



**The Military Theater Distribution
Network Design Problem**

THESIS

MARCH 2015

Robert R. Craig, MAJ, USA
AFIT-ENS-MS-15-M-137

**DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY**

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENS-MS-15-M-137

THE MILITARY THEATER DISTRIBUTION
NETWORK DESIGN PROBLEM

THESIS

Presented to the Faculty
Department of Operational Sciences
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Operational Research

Robert R. Craig, B.S., M.S.

MAJ, USA

MARCH 2015

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

THE MILITARY THEATER DISTRIBUTION
NETWORK DESIGN PROBLEM

THESIS

Robert R. Craig, B.S., M.S.
MAJ, USA

Committee Membership:

LTC Brian J. Lunday, Ph.D.
Chair

Dr. Ray R. Hill, Ph.D.
Member

MAJ Peter A. Nesbitt
Member

Abstract

We formulate and test a mathematical program to select Air and Sea Ports of Debarcation and intermediate logistical distribution centers, through which we route military supplies over a directed transportation network to meet aggregated weekly demands by military units conducting a steady state contingency operation. The multi-objective model seeks to minimize a weighted combination of the total risk encountered by transported supplies, the total distance traveled by supplies, and the maximum per capita workload supported by transportation assets at a given echelon (i.e., port-to-distribution center versus distribution center to demands). Within our formulation, we account for capacities on arc flows and node throughputs, with the latter enabling the representation of material handling equipment limitations at the ports and distribution centers. For our model, we develop and test an Excel-based decision support tool that invokes the commercial solver CPLEX (Version 12.5) to solve the underlying mixed-integer linear program, and we demonstrate its efficacy on a representative instance. For this instance, we identify extreme points and selected interior solutions on the Pareto efficient frontier and examine the model's sensitivity to selected parameters. We conclude by discussing how the model can account for intra-theater airlift and outline modifications that can account for expected pilferage losses within the distribution system.

Key words: Distribution Network, Facility Location, Supply Chain Design

Acknowledgements

I would like to express deep gratitude to my advisor, LTC Brian J. Lunday, for his direction, support, and patience during my research. I wish to thank Dr. Ray R. Hill and MAJ Pete Nesbitt for their insights and contribution.

Robert R. Craig

Table of Contents

	Page
Abstract	iv
Acknowledgements	v
List of Figures	viii
List of Tables	ix
I. Introduction	1
1.1 Overview	1
1.2 Problem Definition and Problem Statement	8
1.3 Research Objectives	10
1.4 Overview of Remaining Chapters	11
II. Literature Review	12
2.1 Overview	12
2.2 Facility Location Models	12
2.2.1 Set Covering Location Problem	15
2.2.2 Maximal Covering Location Problem (MCLP)	17
2.2.3 p-Center Problem	19
2.2.4 p-Median Problem	22
2.3 Network Flow Models	24
2.3.1 Shortest Path Problem	25
2.3.2 Maximum Flow Problem	27
2.3.3 Minimum Cost Flow Problem	29
2.4 Location and Network Flow Problem	33
III. Methodology	35
3.1 Overview	35
3.2 Model Development	36
3.3 Software Used in Modeling	43
IV. Analysis and Results	44
4.1 Scenario Description	44
4.2 Results	48
4.3 Sensitivity Analysis	56

	Page
V. Conclusion	60
5.1 Conclusions.....	60
5.2 Future Research	61
Appendix A. Notional Nigeria Tactical Scenario	64
Appendix B. VBA Code	72
Appendix C. Story Board	108
Bibliography	109

List of Figures

Figure		Page
1	US Military Classes of Supply	3
2	Army Sustainment Operations.....	3
3	ArmyLogistics Operations	4
4	TSC Organizational Chart	5
5	BSB Organizational Chart	7
6	FSC Organizational Chart	7
7	p-Median Example	23
8	Scenario -based Disposition of Units and Possible A/SPODs in Nigeria [31]	45
9	The Undirected Road Network, $G[N,A]$ [31]	46
10	The Undirected Road Network, $G[N,A]$ and Threat [39]	47
11	Pareto Frontier Representation of the Optimal Solutions for Instances A-G	52
12	Optimal Solution for Instance A [39]	54
13	Optimal Solution for Instance C [39]	54
14	Optimal Solution for Instance B [39]	56
15	Optimal Solution for Instance A with Quadratic Risk Quantification [39]	58

List of Tables

Table		Page
1	w-values for Instances A-G	49
2	Selected Optimal Decision Variable Values, Instances A-G	50
3	w-values for Instances A'-G' with Quadratic Weighting	57
4	Selected Optimal Decision Variable Values with Quadratic Weighting, Instances A'-G'	57
5	Linear and Quadratic Objective Function Component Comparison	58

THE MILITARY THEATER DISTRIBUTION NETWORK DESIGN PROBLEM

I. Introduction

“Amateurs talk about tactics, but professionals study logistics.”

Gen Robert Barrow, USMC [28]

This chapter explains the background for this research. First, in Section 1.1 we describe the significance of logistics in historical campaigns and provide an overview of the Army’s distribution system. Next, in Section 1.2 we formally define the problem and present the problem statement. In Section 1.3 we outline the research objectives. Finally, in Section 1.4 we provide an overview of the remaining chapters in this thesis.

1.1 Overview

The purpose of this research is to facilitate the movement of Army cargo/materiel from large depots through a distribution system to the user level where it is needed for warfighting. The success or failure of military campaigns can provide case studies in logistics. Examples are Hannibal’s crossing of the Alps, Napoleon’s march towards Moscow, Sherman’s March to the Sea during the American Civil War, and the Allied Armies’ movement to Berlin after the successful landings on Normandy beaches. More modern examples include the humanitarian response to the 2010 Haiti earthquake and the International Security Force Assistance (ISAF) movement into the most remote villages of the Hindu Kush. The success of the United States military in the field is purposefully reliant on the joint interdependence each service has on the

other services' capabilities. *Army Doctrine Publication 3-0 (ADP 3-0) Unified Land Operations* explains the Army's involvement in the joint environment. This reliance encompasses functions such as intelligence, operations, and logistics, and it allows the Army to focus efforts on its core tasks rather than expend effort on tasks the other services are well suited to perform. For example, the United States Air Force (USAF) provides "worldwide cargo and personnel airlift, air refueling, and aeromedical evacuation," whereas the United States Navy (USN) conducts "over-the-shore" operations to up-load, transport, and down-load cargo" [15]. As the USAF and USN conduct these core tasks of their respective service, the Army is better able to focus on its core tasks. However, although joint interdependence exists by design, Title 10, U.S. Code requires that each service retain responsibility for the sustainment of forces it allocates to a joint force [15].

The Department of Defense's (DOD) Global Distribution process is a joint endeavor in which each service contributes to distribute cargo/materiel to meet the demands of the warfighter. The United States Transportation Command (USTRANSCOM) is responsible for planning, resourcing and executing the Global distribution process which is the mechanism USTRANSCOM uses to move cargo/materiel from points of origin to an aerial or sea port of debarkation (A/SPOD) [7]. The last segment of the transport of cargo/materiel is called intra-theater movement and is the responsibility of each service. This thesis considers the intra-theater distribution of cargo/material by Army systems to support Army forces after the supplies arrive at an A/SPOD. Cargo/materiel, referred to as classes of military supplies in Army Doctrine Publication 4-0 (ADP 4-0) and shown in Figure 1, will heretofore be referred to as supplies.

Class	Item
I	Subsistence (food) and gratuitous (free) health and comfort items
II	Clothing, individual equipment, tentage, organizational tool sets and kits, hand tools, unclassified maps, administrative and housekeeping supplies, and equipment
III	Petroleum, oil and lubricants (package and bulk): petroleum, fuels, lubricants, hydraulic and insulating oils, preservatives, liquids and gases, bulk chemical products, coolants, deicer, antifreeze compounds, components, additives of petroleum and chemical products, and coal
IV	Construction materials, including installed equipment and all fortification and barrier materials
V	Ammunition of all types: bombs, explosives, mines, fuzes, detonators, pyrotechnics, missiles, rockets, propellants, and associated items
VI	Personal demand items (such as health and hygiene products, soaps and toothpaste, writing material, snack food, beverages, cigarettes, batteries, and cameras—nonmilitary sales items)
VII	Major end items such as launchers, tanks, mobile machine shops, and vehicles
VIII	Medical materiel including repair parts peculiar to medical equipment
IX	Repair parts and components to include kits, assemblies, and subassemblies (repairable or nonrepairable) required for maintenance support of all equipment
X	Material to support nonmilitary programs such as agriculture and economic development (not included in Classes I through IX)
Misc	Water, salvage, and captured material

Figure 1. US Military Classes of Supply

The Army's concept of sustainment involves moving and supporting forces. As explained in ADP 4-0, sustainment is built around three elements; logistics, personnel services, and health services, as shown in Figure 2:

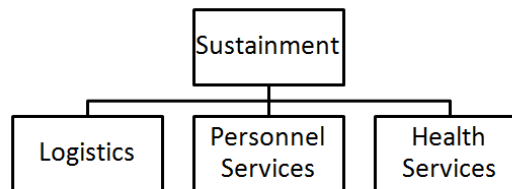


Figure 2. Army Sustainment Operations

The distribution of supplies occurs within the logistics field. Figure 3 from ADP 4-0 displays the tasks Army logisticians plan and execute. This thesis addresses the

distribution of supplies from an A/SPOD through logistic hubs to meet the demands of Army maneuver units in the field.

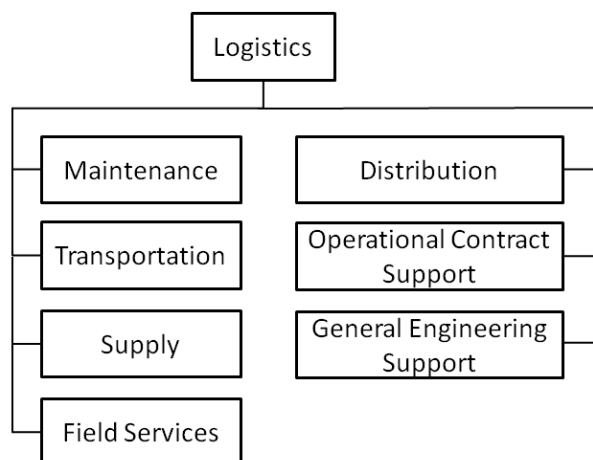


Figure 3. Army Logistics Operations

In order to fight effectively in the field, the Army must conduct unified land operations, a term which describes how the Army fights simultaneously in the offense, defense, and stability operations. To conduct unified land operations the Army must sustain itself to enable operational reach (operating over long distances from an A/SPOD), freedom of action (the depth and breadth of the operational area), and prolonged endurance (extended durations) [16].

The remainder of this section describes the movement of military supplies through five echelons from the A/SPOD to the maneuver companies in the field in order to support unified land operations. However, this research will only consider the movement of supplies from the A/SPOD to brigade combat teams (BCTs) at forward operating bases (FOBs). The five echelons from highest to lowest are the theater sustainment command (TSC), the expeditionary sustainment command (ESC), the sustainment brigade (SB), the brigade support battalion (BSB), and the forward support company

(FSC).

The highest echelon is the Army's TSC, which is a fixed headquarters that provides operational-level support to Army forces or a joint task force (JTF). ADP 4-0 explains the TSCs organizational structure and a diagram is shown in Figure 4. The TSC can support one or more SBs as well as one or more ESCs. The TSC is responsible for distributing supplies from A/SPODs to the next echelon, the ESC.

The second echelon, the ESC, is a headquarters which may either augment a TSC in the event of a very large and enduring operation or, more typically, replace it for a smaller-scale or short-term operation. An ESC is established when multiple SBs are required to support the GCC, or the theater of operations is far away from the TSC such that an additional command node, the ESC, is necessary to facilitate the distribution of supplies. The ESC then becomes a headquarters to SBs.

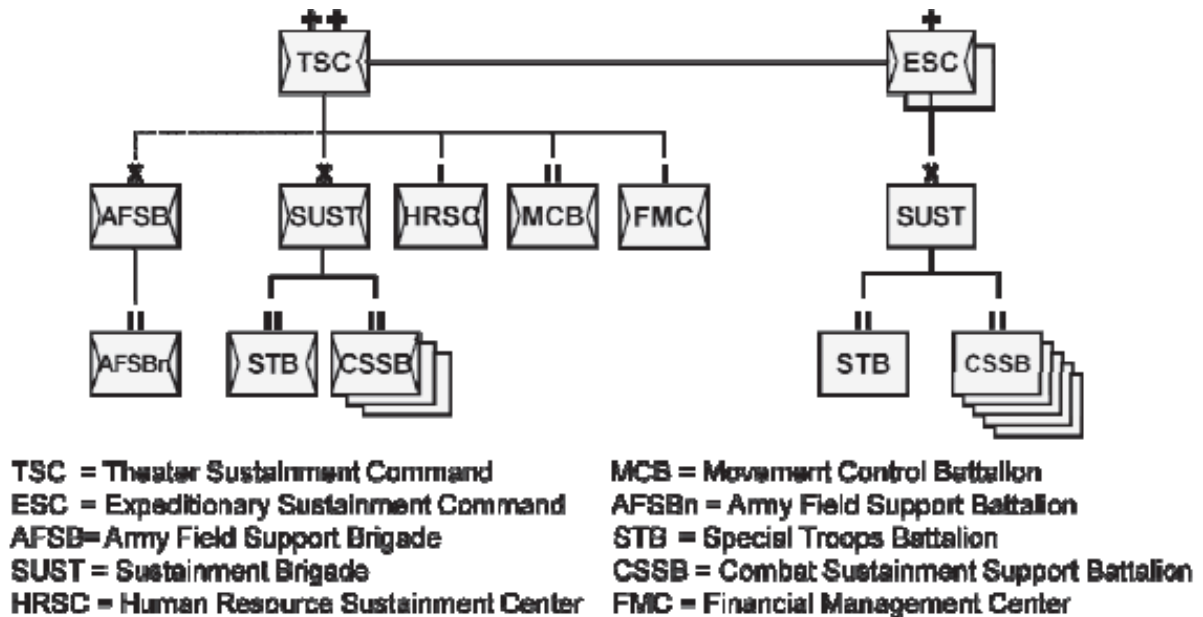


Figure 4. TSC Organizational Chart

The SB is the next organization that is responsible for receiving and moving mili-

tary supplies through the distribution system. The SB supports division-sized units (20,000 soldiers). Like the TSC and ESC, the SB is modular, and its design varies depending on the needs of the GCC. Generally speaking, the SB will have a special troops battalion (STB) and one or more combat sustainment support battalions (CSSBs), as shown in Figure 4. The physical transport of supplies from the TSC/ESC to the SB is accomplished by CSSBs. CSSBs, displayed in Figure 4 are tasked organized with various types of transportation units based on the operational requirements to move supplies within these echelons.

As explained in ADP 4-0, the next organization responsible for movement of military supplies in the distribution system is the BSB, and a representative diagram is shown in Figure 5. The BSB is assigned to the maneuver BCT both in a deployed and non-deployed environment. The relationship between the SB and the BSB is maintained through staff coordination for the requisition and movement of supplies between the two organizations. The SB's support operations officer (SPO) coordinates with the BSB support operations officer (SPO) and the maneuver brigade S3 to move military supplies to the FSCs, the final echelon in the distribution of military supplies. The other units depicted in Figure 5 provide the full complement of support necessary for a BSB to perform the logistics operations depicted in Figure 3. From the BSBs, supplies are transported using internal unit haul assets to the last echelon, the FSCs, each of which supports a battalion -sized unit (~ 600 soldiers).

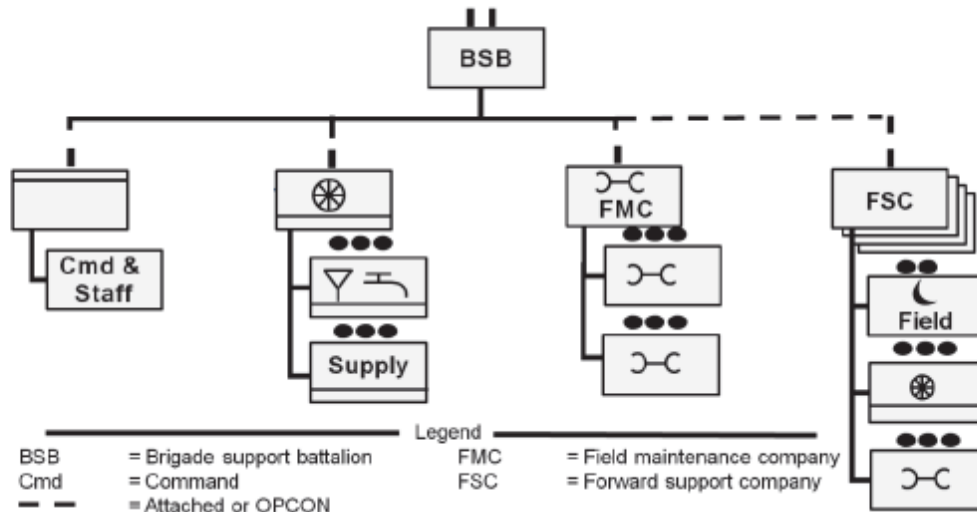


Figure 5. BSB Organizational Chart

Figure 6 from ADP 4-0 shows the FSC organization which moves supplies to the maneuver companies. The FSCs are assigned to the BSB and are under operational control (OPCON) of the supported maneuver battalion. The FSCs move military supplies directly to the maneuver companies. The other units depicted in Figure 6 provide the full complement of support necessary for a FSC to perform the logistics operations depicted in Figure 3.

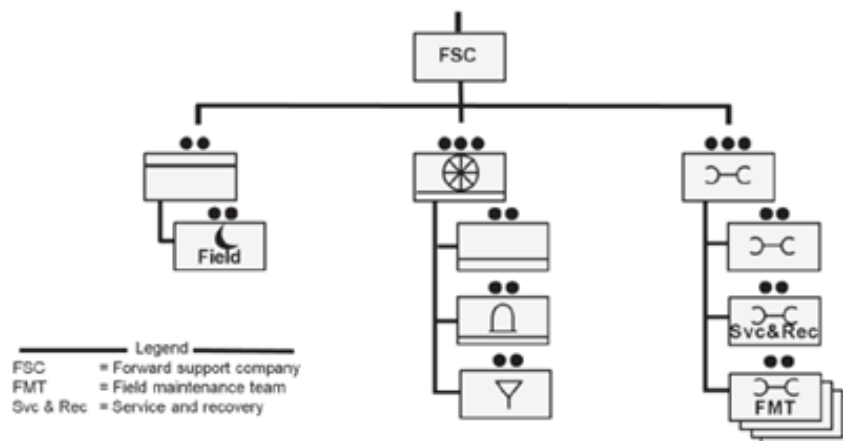


Figure 6. FSC Organizational Chart

This research considers only the flow of military supplies from the TSC/ESC echelon through the SB echelon and to the BSB locations. In terms of physical locations, this movement comprises the flow of supplies from A/SPODs through intermediate distribution centers to Forward Operating Bases (FOBs). Underlying this research scope is the assumption that only a TSC or an ESC is utilized but not both; however, the model we propose may be readily adapted if the assumption does not hold.

1.2 Problem Definition and Problem Statement

Army planners use experience, training center exercises, and models to plan real world deployments. The Army’s Training and Doctrine Command Analysis Center at Ft. Lee, Virginia (TRAC-LEE) is an analysis agency that provides planners with support in planning sustainment operations. This research was initiated to complement TRAC-LEE’s support of United States Army Africa (USARAF) to enhance its logistic planning capabilities, but it could also be used by other major commands (MAJCOM). TRAC-LEE employs a tool called Logistics Battle Command (LBC) that “has planning and decision support features to enable a simulated logistics battle captain to (1) monitor the logistics common operating picture, (2) forecast demand for most classes of supply, and (3) initiate and adjust missions to distribute supplies and perform sustainment functions” [26]. LBC is a dynamic stochastic model that uses demands and nodes to assess the utility of A/SPOD and intra-theater logistic hub locations. TRAC-LEE would be better able to assist USARAF if it was possible to quickly model a logistics infrastructure system that supports operations within its AOR. This research will explore ways to enhance LBC by developing good starting solutions based on analytic rigor.

In support of steady state supply operations for a military contingency operation and

given an existing road network, a set of demands for military supplies, and a finite number of logistical hubs at the various echelons, we seek to locate logistical hubs and route the flow of military supplies to provide effective logistical support. Within this context, we consider alternative means to measure effective logistical support, to include minimizing the total risk encountered by transporting supplies, minimizing the total distance traveled by supplies, balancing the work load between echelons of command or the span of control at each mission command (MC) node. In order to adhere to doctrinal implementation of logistical support operations, we restrict our solutions based on the assumptions that material supplies must transit the established MC echelons (skipping levels is not allowed), the span of control for a headquarters (HQ) is 2-5 subordinate units, and the model is built to support steady-state operations.

Given the preceding doctrinal explanation for the movement of supplies through the Army distribution system, the following is an operational explanation of the movement of supplies that will be used in this research. A TSC or ESC, but not both, will be used. The TSC/ESC will be the coordinating HQ for the movement of supplies from CONUS into the theater. A SB will be co-located at an A/SPOD, and supplies will be moved to regional warehouse distribution centers by CSSB assets. Hereafter, we use the term regional warehouse distribution center (DC) in lieu of the colloquial ‘logistical hubs’ to represent locations established to support the movement, storage, repackaging, and transshipment of military supplies. The DCs are an extension of the SBs and are controlled by the SB at the A/SPOD. In some instances the CSSB may move supplies directly from the SB at an A/SPOD to meet demands at a FOB where a BSB is located. However, in our model this is not allowed as using DCs allows us to seek more efficient methods moving supplies from the A/SPOD to the

FOB. The DC is controlled by a movement control team (MCT) from the SB, and tailored supply packages to meet BCT demands at a FOB are generated by material handling teams (MHE) provided by the SB. Once the tailored supply packages are created, they are moved to a FOB by CSSB assets. Finally, this research will use DCs, but the warehousing aspect of the distribution system will not be considered.

With the above discussion, the following problem statement is proposed. Given an existing capacitated road network, seaports, and airfields; a geographically dispersed set of demands for supplies (i.e., BCT-sized FOB locations); a bounded number of TSC/ESC-operated APODs and SPODs; and a bounded number of SB-operated intermediate distribution centers; we seek to determine good solutions for locating A/SPODs and distribution centers, as well as the routing of supplies over the road network, in order to meet demand. Because ‘good’ is an ill-defined term, we consider three alternative metrics to minimize, either individually or in combination: the accumulated risk to supplies being transported, the total distance traveled by supplies, and the maximum per capita workload supported by transportation assets at a given echelon (i.e., A/SPOD-to-distribution center versus distribution center to demands).

1.3 Research Objectives

There are three proposed research objectives:

1. To develop a math programming model that identifies capacitated A/SPOD locations and intermediate DCs in order to maximize multi-commodity flows over existing in-theater infrastructure networks in support of contingency operations.
2. To solve the model by enabling a simple-to-use-and-understand user-input for critical information about A/SPODs that may be used.

3. To demonstrate the model by using a scenario-based case study.

1.4 Overview of Remaining Chapters

Chapter 2 reviews work that has already been published on the topic of this thesis - facility location multicommodity transshipment problems. A detailed literature search of this topic will help narrow the scope of the research and provide clarity of the research objectives and question. In this way it will be shown that the work in this thesis is original. Chapter 3 presents our modeling assumptions, introduces notation, and formally presents our mathematical program with several variants. It further addresses the limitations of our modeling approach. Chapter 4 provides a description of the model validation using a scenario-based case study. We conclude Chapter 4 with a presentation and comparison of the solutions yielded by our modeling variants. Chapter 5 summarizes the major contributions of this research, addresses some of its limitations, and proposes directions in which to continue and extend the research for even greater utility.

II. Literature Review

“Gentlemen, the officer who doesn’t know his communications
and supply as well as his tactics is totally useless.”

Gen. George S. Patton, USA [25]

2.1 Overview

This chapter provides a review of literature related to selected optimization modeling research. The first part of this chapter provides and discusses problem formulations for several selected facility location models including the set covering problem, the maximal covering location problem, the p-center problem, and the p-median problem. The second part of this chapter provides and discusses problem formulations for several common network flow models including the shortest path problem, the maximum flow problem, and the minimum cost flow problem. Three additional network flow models are discussed as well: the transportation problem, the assignment problem, and the transshipment problem. The last section is a review of the facility location and network flow problem. This review is not exhaustive; rather, it provides the background to highlight and understand the work that has previously been accomplished in these areas.

2.2 Facility Location Models

In general, facility location is a process whereby a commodity is placed in a specific location because the commodity provides an advantage to the decision maker. The location decision is based on conditions in the environment such that the the most benefit is derived. For example, early settlers in the eastern United States established

their communities in locations that provided the most benefit in terms of physical protection, water sources, agricultural opportunities, and transportation systems. Today, retail companies build distribution centers and stores in locations that meet demands and maximize their profits. When location problems become complex, as in the case of designing a supply chain for a contemporary retail company, mathematical programming models can suggest solutions.

One of the first to consider facility location models was Hakimi [22]. He analyzed a network from the perspective of the most geographically centered node. From the central node he calculated the distance to each node in the network and summed the distances. He then designated an adjacent node as the “central” node and repeated the process until all nodes in the network had been selected as the “central” node. He compared all the iterations and noticed that the shortest summation was not always the node in the geographic center. His work essentially became what is today defined as the p-median problem.

Hakimi’s successors (Geoffrion & Graves [20] and Tcha & Lee [37]) noted, that by moving away from the center of the network to other locations within the network, it was possible to maximize and minimize different measures of demand. By considering alternative assumption-based objectives and constraints, several facility location models were developed. In this literature review, we will address the set covering location problem (SCLP), the maximal covering location problem (MCLP), the p-median problem (PMP), and the p-center problem (PCP). While each model can be used to identify an optimal solution to a problem, each differs with regard to its underlying assumptions and constraints.

In a facility location model, the feasible region for facility location can be modeled as either discrete or continuous. A map of an area that a distribution network covers will depict rivers, lakes, roads, road junctions, railroad tracks, built up areas, and other map features. In a ground transportation distribution network, the road junctions are nodes. In a discrete facility location problem there are a finite number of locations to locate a facility. In other words, a facility can be located only at a node. In a continuous facility location problem, a facility can be located in one of two ways. First, it can be restricted to any node or anywhere along an arc (a retailer for a chain of stores). Secondly, if the network structure is considered to be irrelevant, then the facility can be located anywhere in \mathbb{R}^2 (a cell phone tower). This research will consider only discrete models.

The purpose of a facility in a supply chain network is to ship, store, or receive goods. The capabilities of a given facility affect problem formulation and ultimately the optimal solution. A source node sends goods through the network to a sink node which requires goods. All nodes between the source node and the sink node are potential facility locations with the capability to *temporarily* store goods. In an uncapacitated facility location problem (UFLP), the upper temporary storage limit is determined “without any budgetary, technological, or physical restrictions.” Alternatively, the capacitated facility location problem (CFLP) does incorporate an upper temporary storage limit. In both the UFLP and the CFLP each node’s temporary storage limit may not necessarily be the same [41]. This research will incorporate aspects of CFLP, in modeling.

Sometimes in a network there is a reason to give preference to the use of a facility over others, or preference to the service of a given demand over other demands. Per-

haps using a specific facility will result in a tax credit or a public relations advantage. Maybe the cost of using that facility is cheaper than the other facilities and it is best to maximize the use of the lowest cost facility first before maximizing the use of any other facility. Whatever the reason for wanting to use one facility over another, weighting a facility will accomplish give one facility preference over another facility. By weighting a facility, the model will seek to utilize the most heavily weighted facility first. By weighting a demand, the model will seek to utilize the most heavily weighted demand first. This research will consider only weighted models.

2.2.1 Set Covering Location Problem.

The SCLP is focused on minimizing the number of facilities based on specific “cover” criteria to meet demands. The term cover refers to a minimum or maximum allowable distance or time that must be met. In an SCLP, each demand must be covered by a facility. The objective in an SCLP is to cover all of the demands using the minimum number of facilities [23].

Model Development

The following notation is used.

Set Notation

- I = the set of demand nodes indexed by i .
- J = the set of facility locations indexed by j .
- $N_i = \{j | d_{ij} \leq D_c\}$ = the set of nodes j within distance or time D of node i ;
these are the nodes eligible to house facilities which “cover” node i .

Parameters

- d_{ij} = the distance between demand node i and location j .

- D_c = a distance or time standard; a facility sited at some node j within the standard of a demand node i is eligible to serve the demand node.

Decision Variables

- $x_j = \begin{cases} 1 & \text{if we locate at site } j \\ 0 & \text{if not} \end{cases}$

With the given set notation, decision variables, and parameters the SCLP formulation is shown below.

$$\text{Minimize } \sum_{j \in J} x_j \tag{1}$$

$$\text{such that } \sum_{j \in N_i} x_j \geq 1, \quad \forall i \in I, \tag{2}$$

$$x_j \in \{0, 1\}, \quad \forall j \in J. \tag{3}$$

The objective function (1) minimizes the total number of facilities needed. Constraint (2) ensures that every demand in the system is covered by at least one facility. Constraint (3) restricts the decision variable to binary values.

One of the first set covering problems involved the location of emergency service facilities. Toregas *et al.* [38] asked the question, “where should city planners have located emergency service facilities so that all households (the demand) had equal access to coverage?” The critical consideration was the maximum time or distance a household was from an emergency care facility. For example, a household could be located no more than 20 minutes from an emergency service facility, or a household could be located no more than 15 miles from an emergency service facility. There was an

assumption that every household must be covered, and so a minimal but unbounded number of facilities would be built to meet the demand.

The weighted set location covering problem (WSCLP) incorporates a weight h_j that allows the decision maker to give preference or penalty to potential facility locations. The notation for the WSCLP and the constraints are the same as the SCLP, but the objective function would change as shown below:

$$\text{Minimize } \sum_{j \in J} h_j x_j \quad (4)$$

If we capacitate an SCLP, it becomes a type of assignment problem, which we will discuss in Section 2.2.3.

2.2.2 Maximal Covering Location Problem (MCLP).

The MCLP is related to the SCLP. The differences are that the MCLP requires that all demand nodes must be covered and the number of facilities to be located is fixed.

Model Development

The following additional notations beyond those designated in the SCLP formulation are described below.

Decision Variables

$$\bullet \ z_i = \begin{cases} 1 & \text{if demand node } i \text{ is covered} \\ 0 & \text{if not} \end{cases}$$

Parameters

- h_i = the demand at node i .
- p = the number of facilities to locate.

With the given set notation, decision variables, and parameters the MCLP formulation is shown below.

$$\text{Maximize } \sum_{i \in I} z_i \quad (5)$$

$$\text{such that } \sum_{j \in N_i} x_j - z_i \geq 0, \quad \forall i \in I, \quad (6)$$

$$\sum_{j \in J} x_j = p, \quad (7)$$

$$x_j \in \{0, 1\}, \quad \forall j \in J, \quad (8)$$

$$z_i \in \{0, 1\}, \quad \forall i \in I. \quad (9)$$

The objective function (5) maximizes the total demand covered. With an MCLP there is no guarantee that every demand node is covered. Constraint (6) ensures that a demand node is not counted as being covered unless it is sufficiently close to a facility. Constraint (7) fixes the number of facilities to be located. Constraints (8) and (9) are binary logical constraints on facility siting decisions and demand node coverage, respectively [23].

Church & Velle [8] used the MCLP algorithm to help a school district with their busing strategy that had a goal to have a certain percentage of its school age population within a certain walking distance of school. With more students within walking distance, the school district could save money on busing capital and operational costs. However, satisfying the goal required more schools be built than the budget would allow. Essentially, the MCLP imposed a budget constraint by limiting the number of schools that could be built.

The weighted maximal covering location problem (WMCLP) uses a weight h_j that allows the decision maker to give preference or penalty to desired locations. Assume that a location equates to a specific city and larger cities are considered more important. In the MCLP all cities are considered equally important. The objective function covers as many cities as possible without regard to city size. In the WMCLP larger cities would be covered before smaller cities. The notation for the WMCLP and the constraints are the same as the MCLP, but the objective function would change as shown below:

$$\text{Maximize } \sum_{i \in I} h_i z_i \quad (10)$$

If we capacitate an MCLP, it becomes a type of assignment problem which we will discuss in Section 2.23.

2.2.3 p-Center Problem.

The PCP requires a fixed number of facilities and minimizes the maximum distance that a demand is from its closest assigned facility.

Model Development

The following additional notations beyond those designated in the SCLP formulation are described below.

Decision Variables

- W = the maximum distance between a demand node and the facility to which it is assigned.
- $y_{ij} = \begin{cases} 1 & \text{if demand node } i \text{ is assigned to a facility at node } j \\ 0 & \text{if not} \end{cases}$

With the given set notation, decision variables, and parameters the MCLP formulation is shown below.

$$\text{Minimize } W \tag{11}$$

$$\text{such that } \sum_{j \in J} x_j = p, \tag{12}$$

$$\sum_{j \in J} y_{ij} = 1, \quad \forall i \in I, \tag{13}$$

$$y_{ij} - x_j \leq 0, \quad \forall i \in I, j \in J, \tag{14}$$

$$W - \sum_{j \in J} d_{ij} y_{ij} \geq 0, \quad \forall i \in I, \tag{15}$$

$$x_j \in \{0, 1\}, \quad \forall j \in J, \tag{16}$$

$$y_{ij} \in \{0, 1\}, \quad \forall i \in I, j \in J. \tag{17}$$

The objective function (11) minimizes the maximum distance between a demand node and its closest open facility. Constraints (12) and (13) designate how many facilities will be located and ensure that each demand node is assigned to exactly one facility, respectively. Constraint (14) allows node assignments to open facilities only. Constraint (15) sets a lower bound for the maximum distance that is being minimized. Constraint (16) restricts the decision variable to binary values. Constraint (17) requires the demand at a node to be assigned to one facility only [23].

The PCP was developed and solved by Hakimi [22]. The PCP is similar to the SCLP and the MCLP in that they all have covering criteria that drive the objective function. The PCP and the SCLP both require that all demands are covered by a facility, but only the PCP requires a fixed number of facilities. The PCP and the MCLP both require a fixed number of facilities, but only the PCP requires that all demands are

covered by a facility, albeit with a more permissible definition of coverage.

Mumphrey *et al.* [32] advanced Hakimi’s work in this area, but called it “Locating Controversial Facilities.” As an example, consider a ground distribution trucking company that wants to minimize the longest distance any of its trucks travels. The company can locate only p facilities, and it wants to place them such that the furthest distance any delivery truck travels is as short as possible. The PCP can be thought of as an “equity” approach to facility location problems, meaning that the decision maker desires that no truck travels farther than another truck, and will locate facilities to minimize the greatest distance any individual vehicle travels.

The weighted p-center problem (WPCP) uses a weight h_{ij} that allows the decision maker to give preference or penalty to demands. The notation for the WPCP and the constraints are the same as the SCLP, but constraint (15) would change as shown below:

$$W - \sum_{j \in J} h_{ij} d_{ij} y_{ij} \geq 0, \quad \forall i \in I. \quad (18)$$

The capacitated p-center problem (CPCP) supposes that for a given facility at j , it can support at most n demand locations. Accordingly, the CPCP would add the following constraint:

$$\sum_{i \in I} y_{ij} \leq n, \quad \forall j \in J. \quad (19)$$

2.2.4 p-Median Problem.

The PMP requires a fixed number of facilities and minimizes the average distance between the facilities and the demands. The notation for the PMP is identical to that introduced for the PCP, and the PMP formulation is shown below.

$$\text{Minimize } \sum_{i \in I} \sum_{j \in J} d_{ij} y_{ij}, \quad (20)$$

$$\text{such that } \sum_{j \in J} x_j = p, \quad (21)$$

$$\sum_{j \in J} y_{ij} = 1, \quad \forall i \in I, \quad (22)$$

$$y_{ij} - x_j \leq 0, \quad \forall i \in I, j \in J, \quad (23)$$

$$x_j \in \{0, 1\}, \quad \forall j \in J, \quad (24)$$

$$y_{ij} \in \{0, 1\}, \quad \forall i \in I \forall j \in J. \quad (25)$$

The objective function (20) minimizes the average distance between demands and assigned facilities. Constraints (21) and (22) designate how many facilities will be located and ensure that each demand node is assigned to exactly one facility, respectively. Constraint (23) allows node assignments only to open facilities. Constraint (24) restricts the decision variable to binary values. Constraint (25) requires the demand at a node be assigned to only one facility [23].

Current *et al.* [11] provide an application of a PMP using 20 demand nodes labeled A, B,...,T with 10 of those nodes considered possible intermediate locations, namely A, B,...,J, as shown in Figure 7 [11]. In this problem one of the constraints called for

exactly two facilities to be built such that when all demands were routed to one of the two facilities, the sum of the distances was minimized.

The optimal solution is to place nodes at B and J to minimize the total length of the arcs. The solution can be understood graphically by examining Figure 7. The optimal solution was 5133 units when nodes B and J were selected. [Note: [11] did not provide units to his research so the generic term “units” is used.] The pairing with the closest total distance traveled to nodes B and J was B and H at a distance of 6151 units. Placing the facilities at nodes F and J resulted in a distance of 116,410 units which was the maximum distance for this network.

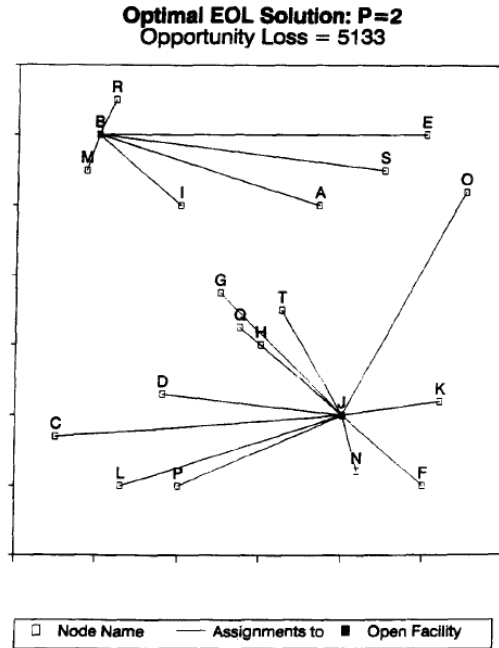


Figure 7. p-Median Example

The weighted p-median problem (WPMP) uses a weight h_{ij} that allows the decision maker to give preference or penalty to desired demands. The notation for the WPMP

and the objective function would change as shown below:

$$\text{Minimize } \sum_{i \in I} \sum_{j \in J} h_i d_{ij} y_{ij} \quad (26)$$

The capacitated p-median problem (CPMP) supposes that, for a given facility at j , it can support at most n demand locations. The CPMP would add the following constraint:

$$\sum_{i \in I} y_{ij} \leq n, \quad \forall j \in J. \quad (27)$$

2.3 Network Flow Models

In 1954 T.E. Harris and F.S. Ross formulated one of the first known flow problems. In 1956 Dantzig, G.B., L. R. Ford, and D.R. Fulkerson furthered this work by developing the first algorithm to compute the maximum flow through a network [13]. This initial work opened exciting opportunities in the discipline of network modeling and was followed by a more than a decade of advancements in this field. Three of the more notable advancements were by Bellman [5] with the SPP, Dijkstra [17] with another SPP algorithm, and Dantzig's work on minimum cost flow models [12].

Network flow models are used to find the most efficient path for moving a commodity from a source node to a sink node through a group of interconnected transshipment nodes. The goal is to move these goods across the network to satisfy the most demand at the cheapest cost, the shortest path, or the maximum flow. A network has at least one source node (supply node) and at least one sink node (demand node). Goods flow through the network from the source node to the sink node. At the source node

the amount of flow out of the node is always greater than the amount of flow into the node. The opposite is true at the sink node. Any other node that may exist in the network is a transshipment node. The amount of flow into a transshipment node is equal to the amount of flow out of a transshipment node [24]. Networks exist in the fields of transportation, communications, manufacturing, power distribution, resource management, financial planning, and many others.

The study of network flow models in effect addresses three questions [1].

1. “What is the best way to traverse a network to get from one point to another as cheaply as possible?”
2. “If a network has capacities on arc flows, how can we send as much flow as possible between two points in the network while honoring the arc flow capacities?”
3. “If we incur a cost per unit flow on a network with arc capacities and we need to send units of a good that reside at one or more points in the network to one or more other points, how can we send the material at a minimum possible cost?”

These questions motivate three of the most common types of network flow models: the shortest path problem (SPP), the maximum flow problem (MFP), and the minimum cost flow problem (MCFP).

2.3.1 Shortest Path Problem.

In this section, we present and discuss the formulation for the SPP. Although several other versions of the SPP exist, this model is a simple network with one source and one sink. More complex models with multiple sources and multiple sinks are not covered.

Model Development

The following notation is used.

Set Notation

- A = the set of m directed arcs.
- N = the set of n nodes.
- $G = (N, A)$ the underlying network.
- I = the set of nodes indexed by i .
- J = the set of candidate facility locations indexed by j .

Decision Variables

- x_{ij} = a binary decision variable that reflects whether arc (i, j) is identified as being on the shortest path from s to t .

Parameters

- c_{ij} = the arc length or arc cost from node i to node j .
- s = the source node.
- t = the sink node.

With the given set notation, decision variables, and parameters the SPP formulation is shown below.

$$\text{Minimize } \sum_{(i,j) \in A} c_{ij}x_{ij}, \quad (28)$$

$$\text{such that } \sum_{j:(i,j) \in A} x_{ij} - \sum_{j:(j,i) \in A} x_{ji} = \begin{cases} 1 & \text{for } i = s \\ 0 & \text{for all } i \in N \setminus \{s, t\} \\ -1 & \text{for } i = t \end{cases} \quad (29)$$

$$x_{ij} = \{0, 1\}, \quad \forall (i, j) \in A. \quad (30)$$

The objective function (28) minimizes the summation of arc flows x_{ij} times their costs c_{ij} . The parameter c_{ij} can represent either a distance or a monetary value. Constraint (29) is commonly referred to as *flow conservation* or *nodal balance* equations. This constraint requires that one unit of flow begins at s and ends at t . Constraint (30) is a binary variable that requires x_{ij} to be a 1 or 0.

The most basic application of the SPP is using a road map to move from a start location to an end location. Although many $s - t$ paths may exist, there is only one shortest length for an $s - t$ path (such a shortest length path may correspond to more than one $s - t$ path, as in the case of alternative optima). Beasley & Christofides [4] used an integer programming formulation to solve a “resource-constrained SPP.” A traveler was given a budget of various resources and was required to reach a given destination as quickly as possible. The various resources were consumed along the path. He had to reach his destination within the resource constraints of his budget. Beasley & Christofides [4] used a lagrangean relaxation to find the shortest path and satisfy the budget constraints.

2.3.2 Maximum Flow Problem.

In this section, we present and discuss the formulation for the MFP. The MFP sends as much flow as possible through a network between a source node s and a sink node t without exceeding the capacity of the arcs.

Model Development

The following notation is used.

Set Notation As defined for the SPP.

Decision Variables

- v = a scalar variable representing the value of the maximum $s - t$ flow.
- x_{ij} = the flow from i to j .

Parameters

- s and t remain as defined in the SPP.
- u_{ij} = the non-negativity flow capacity for arc (i, j) .

With the given set notation, decision variables, and parameters the MFP formulation is shown below.

$$\text{Maximize } v \tag{31}$$

$$\text{such that } \sum_{j:(i,j) \in A} x_{ij} - \sum_{j:(j,i) \in A} x_{ji} = \begin{cases} v & \text{for } i = s \\ 0 & \text{for all } i \in N \setminus \{s, t\} \\ -v & \text{for } i = t \end{cases} \tag{32}$$

$$0 \leq x_{ij} \leq u_{ij}, \quad \forall (i, j) \in A \tag{33}$$

The objective function (31) maximizes the amount of flow v through the network. At the start node s , constraint (32) ensures that the amount of flow out of the node is greater than the amount of flow into the node. Constraint (32) also ensures that at the end node the amount of flow in to the node is greater than the amount of flow out of the node. This constraint also ensures that at every transshipment node in the network the flow into the node is equal to the amount of flow out of the node. Constraint (33) ensures that the flow x_{ij} will always be non-negative.

The MFP is used to model the flow of petroleum products through a pipe network, cars in a road network, messages in a telecommunication network, and electricity in an electrical network. The MFP models arc capacities but not arc costs, whereas the SPP models arc costs but not arc capacities; they are complementary. Together, the properties of the MFP and SPP make up the basis for network flow analysis and lead to the minimum cost flow model described in the next section [1].

The Ford-Fulkerson method (1956) was the first algorithm recorded for computing maximum flow. In 1970 Jack Edmonds and Richard M. Karp developed a more efficient maximum flow algorithm. The Edmonds-Karp algorithm is identical to the Ford-Fulkerson algorithm with one exception. Edmonds and Karp pointed out that “an improper choice of flow augmenting paths can lead to severe computational difficulties.” Edmonds and Karp showed that “if each flow augmentation is made along an augmenting path having a minimum number of arcs, then a maximum flow in an n -node network” can be obtained more quickly [18].

2.3.3 Minimum Cost Flow Problem.

The SPP is a special case of the MCFP, and the MCFP is closely related. The MCFP seeks to “determine a least cost shipment of a commodity through a network in order to satisfy demands at certain nodes from available supplies at other nodes” [1].

Model Development

The following notation is used.

Set Notation As defined in the SPP.

Decision Variables

- x_{ij} = the flow from i to j .

Parameters

- b_i = a variable indicating the amount of supply at node i , where $b_i > 0$ indicates a supply node, $b_i < 0$ indicates a demand node, and $b_i = 0$ indicates a transshipment node.
- c_{ij} = the cost of a unit of flow from i to j .
- u_{ij} = the upper bound on the flow from i to j .

With the given set notation, decision variables, and parameters the MFP formulation is shown below.

$$\text{Minimize } \sum_{(i,j) \in A} c_{ij}x_{ij} \quad (34)$$

$$\text{such that } \sum_{j:(i,j) \in A} x_{ij} - \sum_{j:(j,i) \in A} x_{ij} = b_i \quad \forall i \in N, \quad (35)$$

$$0 \leq x_{ij} \leq u_{ij}, \quad \forall (i,j) \in A. \quad (36)$$

The objective function (34) minimizes the cost of the flow through the network. Constraint (35) ensures a balance of flow at each node. Constraint (36) ensures that the flow x_{ij} will always be non-negative and bounded.

The MCFP is used to model product distribution from the location of product creation to a storage facility such as a warehouse; distribution from warehouses to retailers; the flow of raw materials used in an industrial process as they move through various stations in a production line; and the flow of cars through a network of city streets. The MCFP algorithms are not as efficient as the SPP and MFP algorithms, but the MCFP algorithms are more versatile because they address both flow costs and capacities. Fulkerson & Harding [19] used a MCFP algorithm to maximize the minimum source-sink path subject to a budget constraint. With a fixed budget for

each expenditure on arcs of the network, they allocated the budget among the arcs so as to maximize the length of the shortest path from source to sink” [19].

Of note, there are three special cases of the MCFP (other than the SPP) commonly found in the literature: the general transportation problem, the assignment problem, and the transshipment problem.

The transportation problem is identical to the MCFP with the following exceptions. First, the set of nodes N is divided into two sets of nodes (possibly unequal), N_1 and N_2 , N_1 is a set of supply nodes and N_2 is a set of demand nodes. No transshipment nodes are allowed. Hillier & Lieberman [24] states that “all arcs are directed from a supply node to a demand node, where distributing x_{ij} units from source i to destination j corresponds to a flow of x_{ij} through arc $i \rightarrow j$ ”. The transportation problem does not impose an upper bound on x_{ij} , therefore $u_{ij} = \infty$. The distribution of goods from warehouses to customers is an example of a transportation problem where N_1 represents the warehouses, N_2 represents the customers, and an arc (i, j) represents a path from warehouse i to customer j [1].

Like the transportation problem, the assignment problem is identical to the MCFP with some exceptions. The assignment problem divides the set of nodes N into two sets of nodes, N_1 and N_2 , which represent supply and demand nodes, respectively. However, the cardinality of the two new sets must be equal. Also each supply node has non-negative flow and each demand node has a negative flow. The assignment problem seeks to create at a minimum cost a collection of pairs, one from each set. The pairings must be one-to-one. The most common examples of assignment problems are assigning jobs to machines, students to a computer, runners to a lane at a

track meet, and aircraft to a runway [24] and [1] .

The last special case of the MCFP is the transshipment problem. One difference between the transshipment problem and the MCFP is that the transshipment problem does not have finite arc capacities. For the transshipment problem, an upper bound on the amount of goods that can flow across an arc is limitless; each arc can carry any desired amount of flow required to meet demands [24]. Like both the transportation and assignment problems, the transshipment problem has two sets of nodes, N_1 and N_2 , which represent supply and demand nodes, respectively. Any node that is not within N_1 or N_2 is called a transshipment node. A transshipment node is an intermediate node used as a temporary stopping location for goods flowing from a source node to a sink node. Goods may flow to a transshipment node instead of directly from a source node to a sink node because it is less costly or out of necessity. For example, consider a set of goods leaving a source node bound for three locations via train. The first location is on an island. The second location is in a remote region that does not have a rail system. The third location has a rail system, but needs only a relatively small amount of goods. To move goods to the first location, either a boat or an airplane is required. To move goods to the second location an airplane could be used, but it is most likely cheaper to use ground transportation. Goods to the third location could be delivered via train, but it may be cheaper to use ground transportation due to the relatively small demand. To meet the demands of all three locations, goods should flow into a transshipment node first and then switch to the most economical mode of transportation.

2.4 Location and Network Flow Problem

The models previously described are well defined and can be formulated to fit many situations. However, there are scenarios where both facility location and flow must be considered in the same network. Consider a solid waste collection transfer where trash is collected and then distributed. The flow into the substation from the various pick-up routes will have a maximum capacity and so will the transfer station [1]. In these situations using the facility location problem models and network location models to derive a solution is insufficient. Specifically, representing the actual cost of this operation by modeling a single round trip from a facility to a customer would be inadequate [42].

Location and network flow problems are also called location routing problems (LRP). This name is a misnomer because the LRP is not a well defined problem with distinct parameters. An LRP seeks to solve both facility location problems and network flow problems. The LRP accounts for the individual parameters of the facility location problem, the individual parameters of the network location problem and the relationship between the two. The LRP is also able to consider multiple stops throughout the network [33].

The most pertinent field of study to this research is the LRP. In 2014, Prodhon and Prins [36] provided a thorough review of LRP advancements. Much of the earlier research endeavors such as the warehouse location-routing problem (see, e.g., [2, 35]) are limited in scope and not immediately relevant to our problem; they only address the location of one echelon of sustainment and the routing of supplies to meet demand. More closely related is research on two-echelon location-and-routing problems, as initially examined by Jacobsen and Madsen [27] and Madsen [30]. A general form

of their two-echelon location-routing problem (denoted ‘LRP-2E’) has been studied more extensively in recent years (e.g., see [6, 10, 29, 34]). Although the LRP-2E model is the closest model in the literature to our application, it still differs in several key aspects. The LRP-2E model seeks to locate intermediate distribution centers wherein only the highest and lowest echelon locations are fixed, whereas we also seek to locate the highest level echelon (i.e., the A/SPODs). Moreover, the LRP-2E model routes the delivery of supplies via tours. This constraint is appropriate for certain applications such as newspaper or grocery delivery, but the delivery of supplies on a road network for our context is not similarly restricted.

III. Methodology

“Clearly, logistics is the hard part of fighting a war.”

Lt. Gen. E.T. Cook, USMC [25]

This chapter explains the methodology that we use to solve this problem. First, in Section 3.1 we recall the need for a mathematical model to provide support to TRAC-LEE’s logistics studies. Next, in Section 3.2 we set forth a formal presentation of the math programming model. Finally, Section 3.3 discusses the software used to implement the model and solve the problem.

3.1 Overview

Recall that TRAC-LEE uses LBC (a dynamic stochastic simulation) to assess the effectiveness of locations for A/SPODs and intermediate logistics hubs. LBC requires initial solutions which currently are provided by subject matter experts (SMEs). This research seeks to provide TRAC-LEE with analytic starting solutions for logistics hub locations (APODs, SPODs, and DCs).

United States Army Africa (USARAF) and TRAC-LEE are collaborating to develop supply chain models for contingency operations within the United States Africa Command (AFRICOM) theater of operations. TRAC-LEE uses LBC to provide USARAF with supply chain analysis. This research is conducted to assist TRAC-LEE in providing improved support to USARAF. The model will use Nigeria as a sample area of operations (AO). Once the model is validated, it can be applied to any country within AFRICOM.

3.2 Model Development

This research models the movement of supplies through a transportation network as a mixed-integer linear program (MILP). Within the network, various units transport the military classes of supply from CONUS through A/SPODs into the AO, over air and ground lines of communication (A/GLOCs) through logistic hubs managed by regional warehouse distribution centers (DCs), and subsequently over GLOCs to units at forward operating bases (FOBs). Our model seeks to locate a fixed number of SPODs, APODs, and DCs in order to meet the weekly demands for supplies at FOBs and logistic hubs while minimizing one of several considered objectives. As a baseline, we consider the objective of minimizing the total risk of transporting supplies, but we also consider minimizing the total transportation distances for the supplies as well as minimizing the imbalance of per unit capita responsibility for supply transportation distances.

We represent the A/SPOD-to-FOB distribution infrastructure as a directed network, wherein seaports, airports, FOBs, and road junctions correspond to nodes, and roads and intra-theater air transportation routes correspond to directed arcs. Over this network, we aggregate the point-demand for supplies on a weekly basis, and we model the flow of supplies specific to one of two echelon levels: between the A/SPOD and the DCs, and between the DCs and the FOBs.

Assumptions

Underlying our model are five key assumptions. First, we will consider locating a facility only at a node within the network. This means that a facility cannot be placed along an arc or off of the network itself. This assumption limits the complexity of the formulation and is valid for most objective functions considered [14]. Second,

a DC node can be located at any node in the network. This assumption ensures the widest latitude for the decision maker for emplacing DCs within the limits of the first assumption. Third, all arcs are directed. This ensures that we can characterize the flow over an arc by its direction of movement. Fourth, there is an unbounded supply of material from CONUS. This assumption is used because we are concerned both with the ability of in-theater infrastructure and logistics nodes to meet demands. Finally, we will ship only the amount of supplies needed to meet the demands. This implies that we will not ‘warehouse’ supplies. This research models the sustained conditions after initial deployment into the area of operations when steady-state conditions have been achieved. By limiting flow to exactly meet demands, we can obtain a more accurate assessment of how supplies flow from an A/SPOD to a FOB instead of from a warehouse in the network to a FOB.

Set Notation

- N : the set of nodes in the network indexed by i .
 - $N_a \subseteq N$: the subset of nodes in the network that can serve as an APOD.
 - $N_s \subseteq N$: the subset of nodes in the network that can serve as an SPOD.
- A : the set of arcs in the network indexed by (i, j) .
- $G[N, A]$: the network.
- E : the echelon of commodity flow (indexed by e) in the network, with $E = \{1, 2\}$ corresponding to flow between the A/SPOD and intermediate distribution centers, and to flow between the intermediate distribution centers and the BSBs, respectively.

Parameters

- b_i : the amount of weekly demand for supplies at node $i \in N$, based on the geographic locations of BCTs.
- b^{dc} : the generic weekly demand for an intermediate distribution center (DC).
- c_{ij}^e : the per unit risk incurred to transport supplies of echelon level e on arc $(i, j) \in A$.
- d_{ij} : the distance from node i to node j , defined for $(i, j) \in A$.
- n_a : the maximum number of APODs being designated for CONUS-into-theater flow.
- n_s : the maximum number of SPODs being designated for CONUS-into-theater flow.
- n_e : the number of medium truck companies assigned to transport supplies at echelon $e \in E$.
- n_{dc} : the maximum number of intermediate distribution centers (DCs) being located.
- U_i^a : weekly upper bound on throughput for an APOD located at node $i \in N_a$. This allows the model to account for limitations on maximum on ground (MOG) capacity of potential APODs. MOGs data can be determined by Air Mobility Command (AMC) surveys of potential APODs. For the purposes of keeping this work unclassified U_i^a : values were developed by subject matter expert input.
- U_i^s : weekly upper bound on throughput for an SPOD located at node $i \in N_s$. This allows the model to account for limitations on berthing capacities of potential SPODs. Berthing capacities can be determined by (SDDC) surveys of

potential SPODs for berthing capacities. For the purposes of keeping this work unclassified U_i^a : values were developed by subject matter expert input.

- U_i^{dc} : weekly generic upper bound on throughput for a DC located at node $i \in N$. This allows the model to account for limitations on repackaging and managing supplies at intermediate distribution centers based on the availability of material handling equipment (MHE).
- u_{ij} : upper bound on weekly flow on arc $(i, j) \in A$.
- w_1, w_2, w_3 : the non-negative weights on the objective function components q_1 , q_2 , and q_3 , respectively.

Decision Variables

1. Location Related

- y_i^a : 1 if an APOD is located at node $i \in N_a$, and 0 otherwise.
- y_i^s : 1 if an SPOD is located at node $i \in N_s$, and 0 otherwise.
- y_i^{dc} : 1 if a DC is located at node $i \in N$, and 0 otherwise.

2. Flow Related

- x_{ij}^e : the weekly flow of supplies at echelon $e \in E$ on arc $(i, j) \in A$.
- z_i^c : the amount of weekly flow passing through node $i \in N$ that is converted from Echelon 1 to Echelon 2. This ‘conversion’ represents the receipt, management, and repackaging of bulk supplies into tailored packages for delivery to BSBs.
- z_i^n : the amount of weekly flow passing through node $i \in N$ that is *not* converted from Echelon 1 to Echelon 2.

3. Intermediate Decision Variables

- q_1 : the total risk accumulated to supplies flowing through the network.
- q_2 : the maximum transportation distance through the network.
- q_3 : the maximum per capita (i.e., per medium truck company) workload supported by transportation assets at a given echelon $e \in E$.

Problem Formulation

Within this framework, we propose the following formulation for the Military Theater Distribution Network Design Problem (**MTDNDP**):

$$\text{Minimize } \sum_{k=1}^3 w_k q_k, \quad (37)$$

$$\text{Subject to } q_1 = \sum_{(i,j) \in A} \sum_{e \in E} c_{ij}^e x_{ij}^e, \quad (38)$$

$$q_2 = \sum_{(i,j) \in A} \sum_{e \in E} d_{ij} x_{ij}^e \quad (39)$$

$$q_3 \geq \frac{1}{n_e} \sum_{(i,j) \in A} d_{ij} x_{ij}^e, \quad \forall e \in E, \quad (40)$$

$$\sum_{i \in N_s} y_i^s \leq n_s, \quad (41)$$

$$\sum_{i \in N_a} y_i^a \leq n_a, \quad (42)$$

$$\sum_{i \in N} y_i^{dc} \leq n_{dc}, \quad (43)$$

$$y_i^s + y_i^a + y_i^{dc} \leq 1 \quad \forall i \in N, \quad (44)$$

$$\sum_{j: (j,i) \in A} x_{ji}^1 = z_i^c + z_i^n, \quad \forall i \in N, \quad (45)$$

$$z_i^c \leq U_i^{dc} y_i^{dc}, \quad \forall i \in N, \quad (46)$$

$$z_i^n = \sum_{j: (i,j) \in A} x_{ij}^1, \quad \forall i \in N \setminus \{N_a \cup N_s\}, \quad (47)$$

$$\sum_{j:(i,j) \in A} x_{ij}^1 \leq z_i^n + U_i^s y_i^s, \quad \forall i \in N_s \setminus \{N_a \cap N_s\}, \quad (48)$$

$$\sum_{j:(i,j) \in A} x_{ij}^1 \leq z_i^n + U_i^a y_i^a, \quad \forall i \in N_a \setminus \{N_a \cap N_s\}, \quad (49)$$

$$\sum_{j:(i,j) \in A} x_{ij}^1 \leq z_i^n + U_i^a y_i^a + U_i^s y_i^s, \quad \forall i \in \{N_a \cap N_s\}, \quad (50)$$

$$\sum_{j:(j,i) \in A} x_{ji}^2 + z_i^c = \sum_{j:(i,j) \in A} x_{ij}^2 + b_i + b^{dc} y_i^{dc}, \quad \forall i \in N, \quad (51)$$

$$\sum_{e \in E} x_{ij}^e \leq u_{ij}, \quad \forall (i,j) \in A, \quad (52)$$

$$x_{ij}^1 \geq 0, \quad \forall (i,j) \in A, \quad (53)$$

$$x_{ij}^2 \geq 0, \quad \forall (i,j) \in A, \quad (54)$$

$$y_i^s \in \{0, 1\}, \quad \forall i \in N_s, \quad (55)$$

$$y_i^a \in \{0, 1\}, \quad \forall i \in N_a, \quad (56)$$

$$y_i^{dc} \in \{0, 1\}, \quad \forall i \in N. \quad (57)$$

The objective function (37) minimizes a non-negatively weighted sum of three measures of performance: the total risk encountered by transporting supplies, the total distance traveled by supplies, and the maximum per capita workload supported by transportation assets at a given echelon. Constraints (38) and (39) calculate the total accumulated risk and traveled distance by units of supply flowing through the network, respectively. Constraint (40) bounds the average distance traveled per truck company transporting supplies for each echelon in the network. Constraints (41), (42), and (43) enforce the respective upper bounds on the number of SPODs, APODs, and DCs that may be emplaced, and constraint (44) restricts location decisions to prevent a co-location of an SPOD, APOD, and/or DC at the same node. Although such a disposition of sustainment nodes may be possible, it is trivial within our context, and so we utilize this constraint to set aside the possibility. Should a reader wish to allow

such a co-location, one need only eliminate the appropriate binary variables from constraint (44), or eliminated the constraint in entirely. Constraints (45)–(51) enforce conservation of flow at each node while accounting for the transition of supplies between echelons at distribution centers. Constraint (45) requires that all Echelon 1 flow coming into a node remains as Echelon 1 flow or is converted into Echelon 2 flow. Constraint (46) ensures that flow may only be converted from Echelon 1 to Echelon 2 at a node if an intermediate distribution center node is emplaced there and, for such a case, it bounds the corresponding weekly converted flow by U^{dc} . For nodes at which neither an APOD nor an SPOD may be located, constraint (47) requires that the amount of Echelon 1 flow into this node that was not converted into Echelon 2 flow will depart the node as Echelon 1 flow. For nodes at which an SPOD but not an APOD may be located, constraint (48) ensures that the amount of Echelon 1 flow departing a node is bounded above by the sum of the non-converted Echelon 1 flow that entered the node and the flow that entered from CONUS if that node is designated as SPOD, where U_i^s represents the limit on weekly throughput into the port. Using a parallel construct, constraint (49) enforces a similar constraint for nodes that may be selected to serve as an APOD but not an SPOD, and constraint (50) does the same for nodes that may be selected to serve as either an SPOD or an APOD. Given the restriction imposed by constraint (44), it may appear reasonable to replace constraints (48)–(49) with constraint (50) indexed over $i \in \{N_a \cup N_s\}$; however, we maintain them within this model for the purpose of extensibility, in the event that a user does not assume that distribution centers cannot be co-located with A/SPODs. For every node, constraint (51) requires that the sum of Echelon 2 flow and converted Echelon 1 flow coming into a node is equal to the sum of the Echelon 2 flow coming out of the node, the demand at that node, and any demand that is induced by locating a distribution center at that node. Constraint (52) imposes an upper limit on

the weekly total of Echelon 1 and Echelon 2 flow for each arc in the network, constraints (53) and (54) ensure that both Echelon 1 and 2 flows are non-negative, and constraints (55)–(57) enforce binary logical constraints on facility location decisions.

3.3 Software Used in Modeling

Two types of software are used in this research: Microsoft Excel (Excel) and IBM ILOG CPLEX Optimization Studio (CPLEX). Excel is a spreadsheet application that allows calculation, graphing, and a macro programming language called Visual Basic for Applications (VBA). CPLEX is a commercial solver for linear, integer, and mixed integer linear programming problems.

We utilize Microsoft Excel to store the sets and parameters underlying a given instance, and we use the embedded VBA capability to represent our MILP for the instance. Also using VBA, we invoke the commercial solver CPLEX to solve our MILP and present our solution to the decision maker.

IV. Analysis and Results

“The line between disorder and order lies in logistics...”

Sun Tzu [25]

This chapter explains the results yielded in this analysis. First, in Section 4.1 we present the scenario which was used to test the modeling tool. Next, Section 4.2 discusses the results using seven weighting variations. Finally, in Section 4.3 we include a sensitivity analysis using one of the three components of the objective function.

4.1 Scenario Description

To test our model, we developed an unclassified representative humanitarian assistance (HA) scenario in the country of Nigeria described in Appendix A. Although any country could be used, we chose Nigeria because it provides the opportunity for selection among multiple A/SPODs, and there was sufficient available data to keep this research unclassified.

USARAF is responsible for strategic, operational, and tactical operations across the full range of military operations for the African continent. Moving military supplies into an AO to meet war fighter demands is critical to USARAF’s success. The scenario provides the background and rationale for deploying troops to Nigeria, and includes a regional overview of neighboring countries and Nigeria, a terrain analysis, a human terrain analysis, and a time line of significant events leading to a deployment of US troops. With a mission to provide humanitarian aid and enable civilian authority, this research concerns the placement of logistic hubs to support this mission.

As illustrated in Figure 8, this scenario requires the design of a theater distribution network to sustain nine BCT-sized units at geographically dispersed FOBs, and we restricted supplies to flow through at most one of five possible APODs and at most one of two possible SPODs. We further restricted the scenario to use at most two intermediate distribution centers. The choice of $n_{dc} = 2$ for use within constraint (43) was based on the number and disposition of units, which correlate to a mission run by a joint task force with two subordinate division headquarters, each of which would have an aligned sustainment brigade operating a regional distribution center. The induced network, $G[N, A]$, was developed using a more detailed road network map from the World Food Programme [43]. Considering highways, primary roads, and secondary roads for the transport of military supplies, the resulting network has $|N| = 135$ and $|A| = 414$. Within this structure, we selected $n_e = 2$ medium truck companies for $e = 1, 2$ based on subject matter expert input [21].

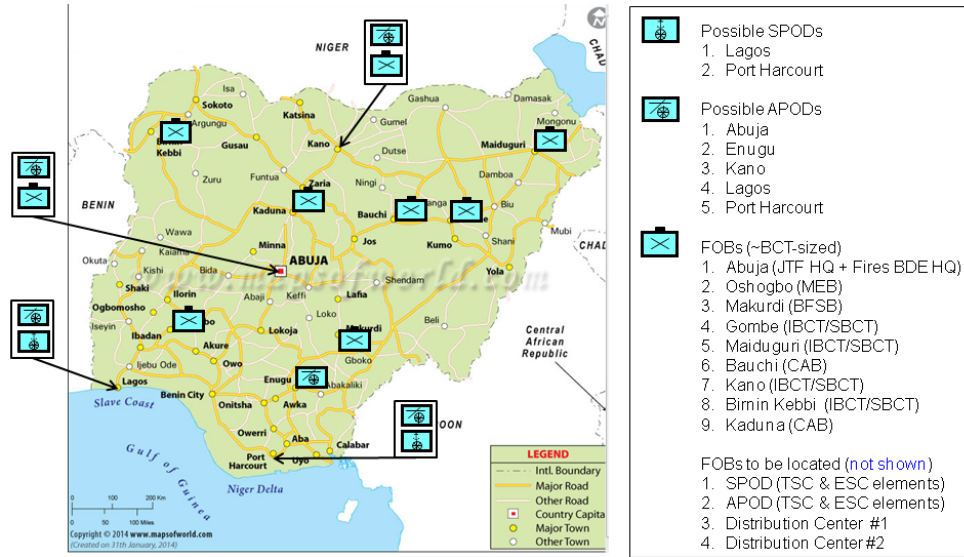


Figure 8. Scenario -based Disposition of Units and Possible A/SPODs in Nigeria [31]

Figure 9 displays the undirected road network used in the scenario. The network is only a portion of the road system across Nigeria. Each of its 135 nodes in the network

which are depicted as blue circles with a corresponding number inside, is located at a road intersection and represent the only locations on the map where facilities can be located. Each arc is depicted as a heavy black line connecting nodes. As stated in Chapter 3, we assume a directed network that ensures that we can characterize the flow over an arc by its direction of movement.

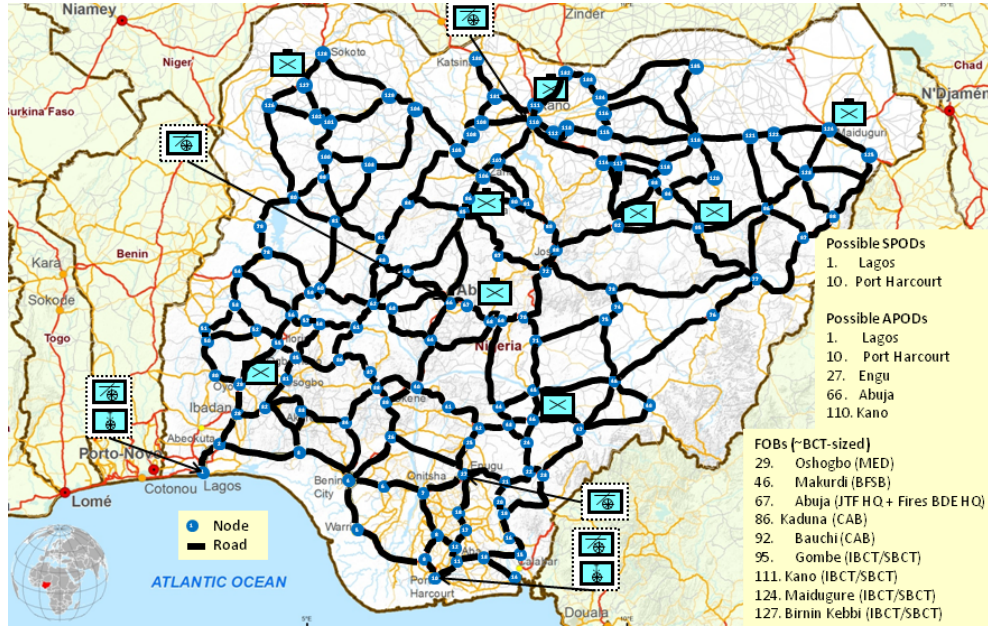


Figure 9. The Undirected Road Network, $G[N,A]$ [31]

We estimated the demand at each FOB (b_i) using a theater sustainment planning reference [9], yielding approximately 285 tons of weekly demand for food, fuel, spare parts, mail, and selected other classes of supply at each such location. Weekly demand at to-be-located DCs (b^{dc}) was estimated at 9 tons of similar supplies. In the absence of specific information about material handling equipment (MHE) restrictions, we set the upper bounds on throughput at APODs (U_i^a), SPODs (U_i^s), and DCs (U_i^{dc}) to be 1000, 3000, and 1800, respectively, so that the scenario would require the use of an APOD, and SPOD, and both DCs. We estimated distances (d_{ij}) to the nearest five meters using a road network map [43], and we assumed upper bounds (u_{ij}) of 1600,

800, and 200 tons for highways, primary roads, and secondary roads, respectively. To determine the risk parameters for each arc, we quantified the qualitative geographic risk assessments within Nigeria by the United Kingdom’s Foreign and Commonwealth Office (FCO) [39]. We used the FCO’s 2012 assessment in lieu of a more recent assessment in order to maintain the notional nature of the scenario, and we weighted the FCO’s categories of “advise against all travel”, “advise against all but essential travel”, and “see our travel advice before traveling” with linearly scaled values of $c_{ij}^1 = c_{ij}^2 = 3, 2$, and 1, respectively. Of note, as shown in Figure 10 each arc’s risk classification was determined by the highest risk category for any region through which it traverses, and we did not assume a different category of risk for supplies being transported at different echelons of flow.

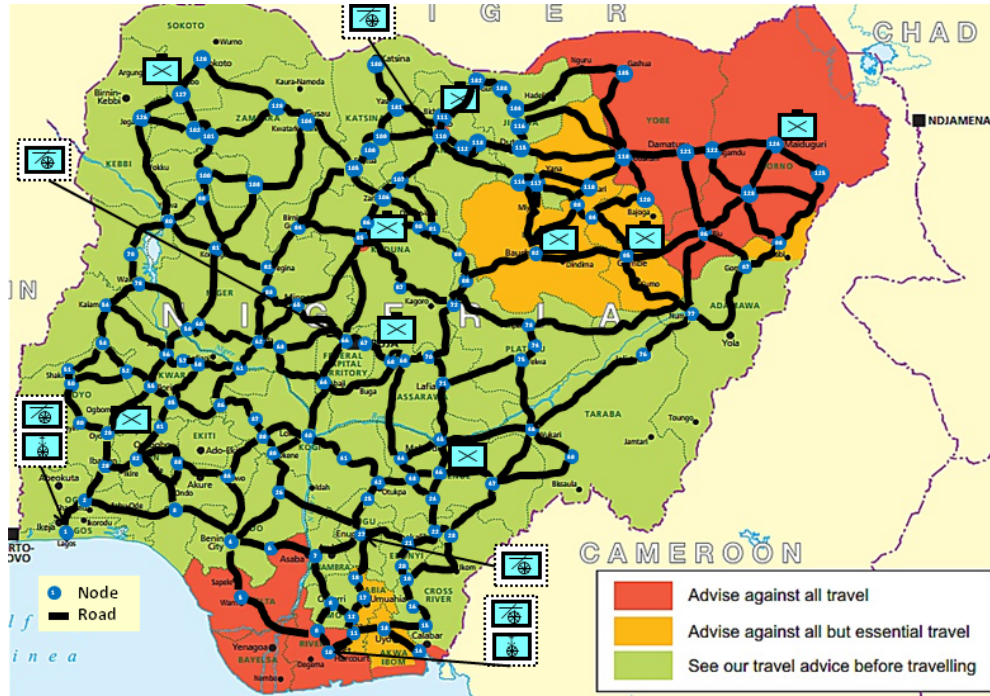


Figure 10. The Undirected Road Network, $G[N,A]$ and Threat [39]

4.2 Results

In this section we analyze results using seven instances with different weighting variations corresponding to a letter (A, B, C, D, E, F, and G). We analyze both the objective function components and the decision variables with respect to each instance. As described in Section 3.2, the objective function (37) consists of three measures of performance; the accumulated risk to supplies flowing through the network (q_1); the total distance traversed by supplies (q_2); and the maximum supply-distance load among the two defined echelons of transportation (q_3). However, it is true that examining the entire objective function value at optimality is not particularly useful. The reason is because the units on q_1 , q_2 , and q_3 are not the same, and those components might not be well scaled. Moreover, a component-wise optimization of the objective function will likely indicate that they are in tension. These reasons indicate a component-wise analysis of the objective function is in order, which we accomplish by examining the efficient Pareto frontier. The terms *risk*, *distance*, and *balance* are used interchangeably for the calculations q_1 , q_2 , and q_3 , respectively.

Because the multiple objectives have different units of measure and different scales, we examined seven different instances for possible corresponding weights, as indicated in Table 1, each of which yields a unique non-dominated solution on the efficient Pareto frontier. To identify distinguishable non-dominated solutions for our three-dimensional Pareto frontier, we first consider three instances (A-C) for the given network, where we weight only one objective function component at a time. Next we consider two objective function component instances. The weights for Instances D-F, a scaled weighting for the combinations of $\binom{3}{2}$ objectives, with w_1 , w_2 , and w_3 are calculated using the weighted sum method based on the optimal solutions to Instances A-C. Finally, we determine the weights for all three components in a similar manner

to generate and solve the three objective function component, Instance G [3].

To determine the two weights, w_i and w_j , we solve the set of equations:

$$(q_i^*)w_i = (q_j^*)w_j. \quad (58)$$

$$w_i + w_j = 1. \quad (59)$$

where q_i^* is the value from the instance (A, B, or C) where $w_i = 1$, and likewise for q_j^* . This weighted sum method ensures the respective weights better address the relative scaling of the objective function components. In this manner, we seek to determine appropriate weights that account for possible poor scaling among the objective function components.

Table 1. w-values for Instances A-G

Instance	w_1	w_2	w_3
A	1	0	0
B	0	1	0
C	0	0	1
D	0.98	0.02	0
E	0.94	0	0.06
F	0	0.23	0.77
G	0.93	0.02	0.05

Given comma-delimited files for the node- and arc-specific data for the scenario's network shown in Figure 10 (or any other instance), we encoded in Visual Basic a program to construct the formulation within Microsoft Excel and, given user-input weights for the respective objectives, invoke the commercial solver CPLEX (Version 12.5) to solve Problem MTDNDP. The average required computational effort for all seven instances was less than two minutes on a computer having an 2.70 GHz AMD Athlon II X2 processor and 4.00 GB of RAM, indicating the efficacy of our formulation and solution method to address larger instances. Reported in Table 2 are selected

elements from the optimal solutions for Instances A-G; the first three columns tabulate the optimal values for the respective objectives corresponding to the accumulated risk, the distance, and the maximum per capita workload supported by transportation assets at a given echelon (balance), and the final three columns indicate the number of the node selected for use as an APOD, an SPOD, and DCs. For the last row of Table 2 we denote RPD_i to be the relative percentage deviation between the worst and best optimal values for q_i , $i=1, 2, 3$, over Instances A-G.

Table 2. Selected Optimal Decision Variable Values, Instances A-G

Instance	q_1	q_2	q_3	APODs	SPODs	DCs
A	21,666	1,536,885	678,868	110	1	2, 107
B	28,457	1,213,825	543,838	110	1	2, 111
C	44,400	1,486,964	371,741	110	10	40, 111
D	22,968	1,400,515	563,183	110	1	2, 114
E	26,136	1,524,649	381,162	110	1	28, 92
F	42,457	1,416,260	384,700	110	10	40, 111
G	25,568	1,450,775	400,425	110	1	28, 92
RPD_i	105%	27%	83%			

Intuitively, each of the respective objectives attained its lowest value when only it was weighted, as is the case for Instances A-C. For example, the weighting for Instance A $(w_1, w_2, w_3) = (1, 0, 0)$ returns the lowest value for q_1^* for all seven instances. With this weighting, only the risk-related objective is forced to minimize. The distance-related and balance-related objectives are allowed to fluctuate in order to minimize the risk-related objective and are not the lowest values for the seven instances. The weighting for Instances B and C return the lowest values for distance-related and balance-related objectives, respectively, in the same manner.

Despite the common use of the APOD at Kano (i.e., Node 110) for all instances, the optimal solution for each instance was unique with regard to other sustainment node

emplacements and/or supply flow routing. Although other non-dominated solutions may exist on the efficient Pareto frontier, we did not seek to find them via an exhaustive search; we intend for this model merely to provide several distinguishable alternative (nondominated) solutions for a decision maker to consider. Of note, the RPD between the worst and best optimal values for q_1^* , q_2^* and q_3^* over Instances A-G were respectively 105%, 27%, and 83%. This result indicates a higher sensitivity of both the risk-related and the workload-balancing-related objectives to variations in the objective weights for this scenario.

Although the RPD for the distance-related objective was only 27%, this result is not insignificant; it corresponds to a possible increase in road mileage, wear-and-tear on vehicles, and fuel usage of up to 27%, depending on the non dominated solution selected for implementation. The RPD for the balance-related objective was 83% between the highest and lowest solutions which corresponds to a significant imbalance between the work load of units designated to haul Echelon 1 and Echelon 2 flow. With such a strong discrepancy in work load, a decision maker may consider changes to the task organization to prevent both personnel and vehicles being over utilized. The RDP for the risk-related objective was 105%. This corresponds to units hauling supplies being as much as twice as vulnerable to a threat depending on the non dominated solution selected.

Figure 11 illustrates the seven non-dominated points on the Pareto frontier corresponding to Instances A-G. the Pareto frontier consists of non-dominated solutions. The continuous surface that includes these points looks somewhat like an irregular-shaped bowl made of facets. We seek to minimize the objective function components. The points closer to the origin (points ‘below’ or ‘outside’ the bowl) are not feasible.

In contrast and in the context of Figure 11, feasible points exist that are closer to the reader (‘above’ or ‘inside’ the bowl). However, these points are dominated by points on the the Pareto frontier because there exists a point on the frontier that has the risk, distance, and balance values at least as low as a dominated solution, with one of the component values being strictly lower.

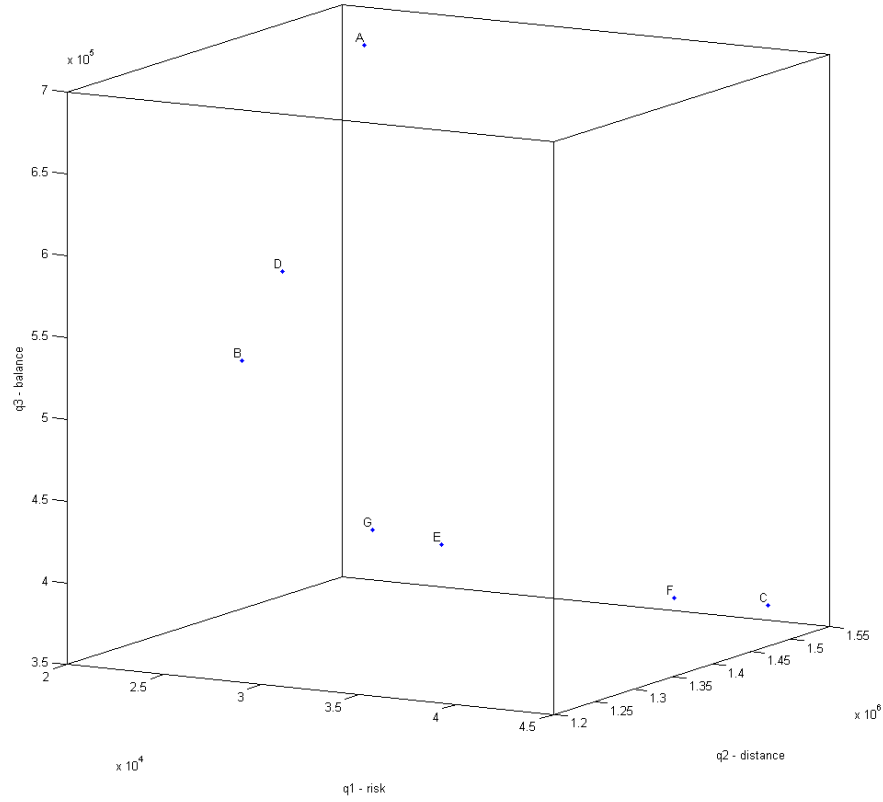


Figure 11. Pareto Frontier Representation of the Optimal Solutions for Instances A-G

Beyond a comparison of the values for the objective function components, there is merit to examining the other optimal objective function values for the seven instances. Such a comparison seeks to identify whether the optimal solutions are distinguishable [40]. Moreover, it may identify commonalities among the optimal solutions as well, thereby generating insights on solution characteristics that are sensitive to relative

component weightings. Thus, while Instances A-G are all on the Pareto frontier, we show that Instances A, B, and C provide the best solutions when considering trade-offs between the risk-related, distance-related and balance-related objectives. Instances D, E, F, or G do not allow for the minimum value of any combination of measure of performance for the the risk-related, distance-related, and balance-related objectives. For example, when comparing Instances D and A, more balance is achieved between Echelon 1 and Echelon 2 flow in Instance D, as well as a lower distance traveled in Instance D. However, Instance A still returns a smaller risk value.

To illustrate elements of the characteristics of our solutions, we depict in Figures 12 and 13, the optimal solutions to Instances A and C, overlaid on a map that depicts the geographic risk assessments, respectively [39]. Within each figure, we depict APODs, SPODs, and DCs as red, orange, and yellow nodes, with all other nodes being blue in color; and we represent Echelon 1 and 2 flows by coloring the arcs on which they are shipped as green and purple, respectively, with some arcs supporting flow at both echelons. Using the graphic solution for each instance along with Table 2, the relationships among the instances can be explored.

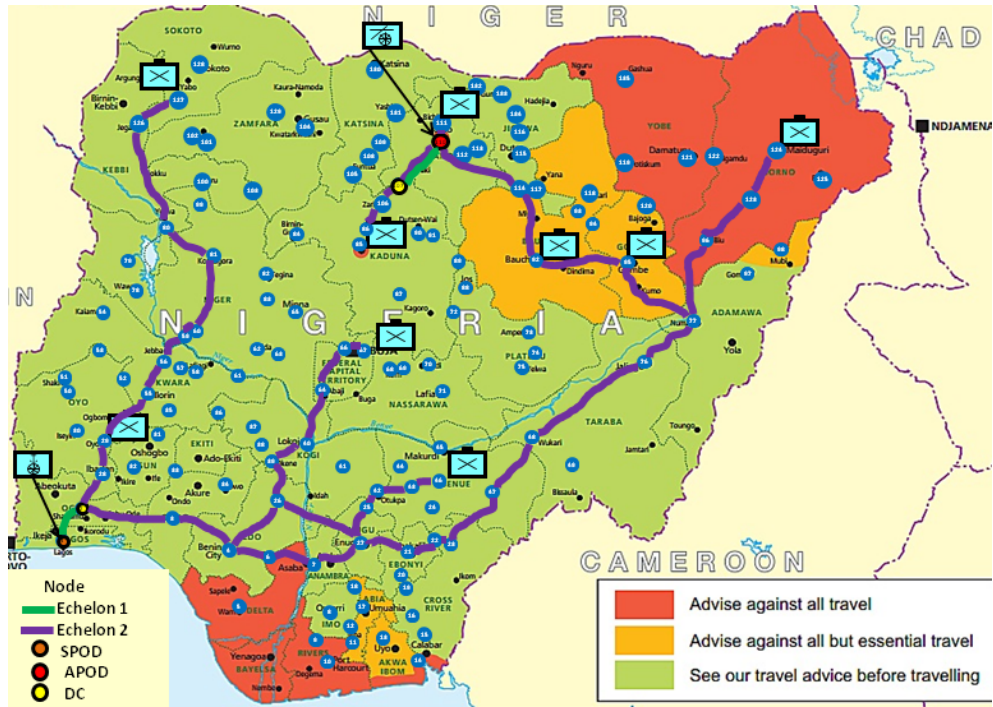


Figure 12. Optimal Solution for Instance A [39]

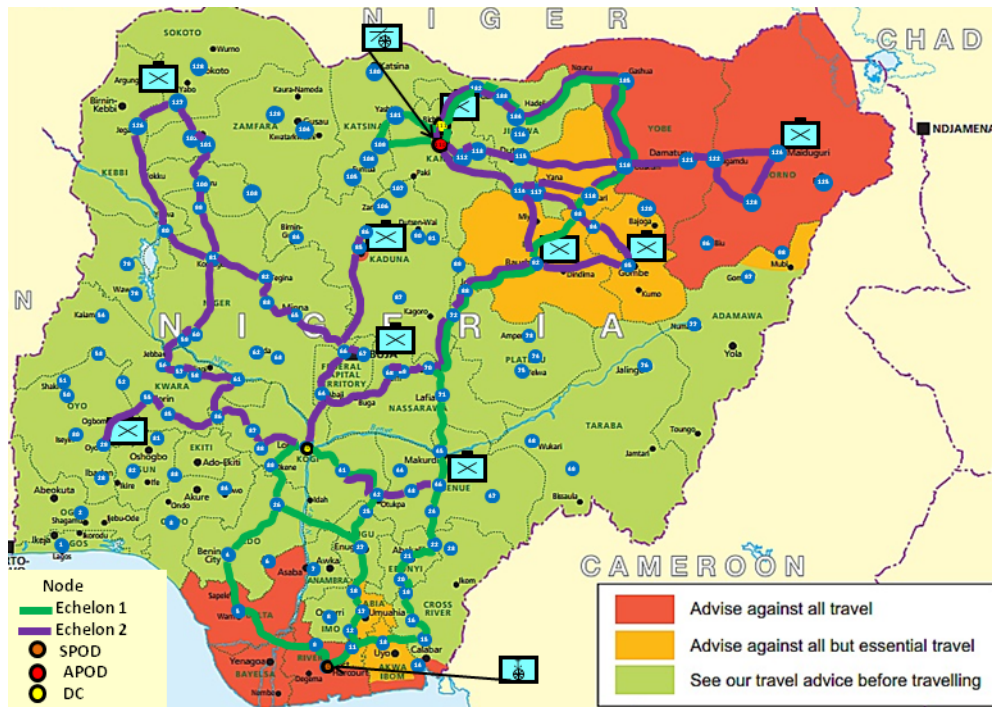


Figure 13. Optimal Solution for Instance C [39]

Visible in Figure 12 is the most risk-averse solution; it utilizes Lagos (i.e., Node 1) as

the SPOD in lieu of Port Harcourt (i.e., Node 10), and it uses very few of the roads in the high threat areas of the country. However, it also emplaces the distribution centers very close to the selected APOD and SPOD, yielding an imbalanced load between the two echelons. The supply-distances supported by the second echelon are 7.5 times as great as the first echelon, indicating that a change in transportation resource allocation will be necessary to implement this solution.

In contrast, the optimal solution for Instance C perfectly balances the workload between the two echelons by emplacing one of the the distribution centers in the more centrally-located town of Lokoja (i.e., Node 40). However, the greatest risk-based objective is attained via the solution to Instance C, wherein Port Harcourt is utilized as an SPOD and flows are routed throughout the high threat areas without regard to the risk incurred to convoys. This graphic finding is supported by Table 2 where the highest value for q_1^* was attained when $(w_1, w_2, w_3) = (0, 0, 1)$ (i.e., Instance C), and the highest value for q_3^* was attained when $(w_1, w_2, w_3) = (1, 0, 0)$ (i.e., Instance A), indicating the tension between these two objectives for this scenario. Moreover, both Instances A and C do not address the distance-based metric, resulting in the respective (ordinally ranked) first and third worst values for q_2^* .

Should a decision maker be concerned only with minimizing the total distance traversed by all supplies, consider the solution to Instance B as depicted in Figure 14 [39]. In order to attain the shortest supply routings, this solution incurs a 31% increase in the risk-based metric compared to Instance A by routing supplies through higher-risk regions, and it allows for a second-echelon workload that is 8.6 times the first-echelon workload by locating the two distribution centers close to the selected APOD and SPOD.

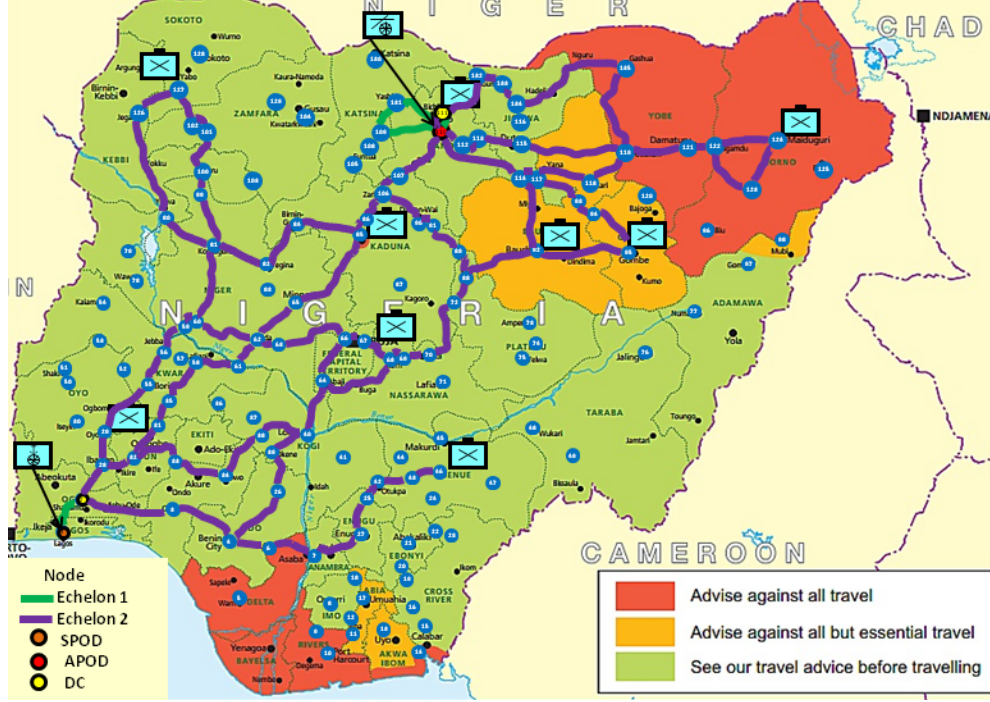


Figure 14. Optimal Solution for Instance B [39]

4.3 Sensitivity Analysis

Acknowledging that our quantification of risk via a linear metric is not the only suitable method, we examined the sensitivity of our solutions to the imposition of a quadratic risk-based metric, with values of $\{1, 4, 9\}$ in lieu of $\{1, 2, 3\}$. Using the same weighted-sum method to determine the objective weights when more than one objective is considered, we calculated the new weights for all seven instances as shown in Table 3.

We then used the same comma-delimited files for the node- and arc-specific data, Visual Basic program, and CPLEX (Version 12.5) solver to obtain optimal solutions for all seven instances as shown in Table 4.

Similar to the linearly weighted optimal solutions, the quadratic risk-based optimal solutions attain their lowest value when only each objective is individually weighted, as in the cases for Instances A'-C'. Also, the placement of APODs, SPODs, and DCs

Table 3. w-values for Instances A'-G' with Quadratic Weighting

Instance	w_1	w_2	w_3
A'	1	0	0
B'	0	1	0
C'	0	0	1
D'	0.98	0.02	0
E'	0.93	0	0.07
F'	0	0.23	0.77
G'	0.91	0.02	0.07

Table 4. Selected Optimal Decision Variable Values with Quadratic Weighting, Instances A'-G'

Instance	q_1	q_2	q_3	APODs	SPODs	DCs
A'	27,374	1,579,285	700,068	110	1	2, 107
B'	49,625	1,213,825	543,838	110	1	2, 111
C'	87,907	1,486,964	371,741	110	10	40, 111
D'	29,157	1,411,490	616,170	110	1	2, 107
E'	35,495	1,529,231	382,308	110	1	28, 92
F'	84,119	1,416,260	384,700	110	10	40, 111
G'	35,778	1,475,800	390,250	110	1	28, 92
RPD _i	221%	30%	88%			

is the same between the linear and quadratic optimal solutions with the exception of Instance D and D'. Differing notably between the linear and quadratic risk-based optimal solutions are the RPD values for each of the objectives. The RPD between the worst and best optimal values for q_1 , q_2 and q_3 over the instances with the alternative risk quantification were 221%, 30%, and 88%, respectively, an increase of 116%, 3%, and 5% over the linear risk-based optimal solutions displayed in Table 2. A comparison of the optimal objective function components for the linear and quadratic models is shown in Table 5. The third column displays the percent change (PC) between the linear and quadratic objectives for each instance

The risk-related objective experienced in some cases almost 100% PC between the linear and quadratic models, indicating that the risk-related objective is highly sen-

Table 5. Linear and Quadratic Objective Function Component Comparison

Instances	q_1			q_2			q_3		
	Lin	Quad	PC (%)	Lin	Quad	PC (%)	Lin	Quad	PC (%)
A, A'	21,666	27,374	26	1,536,885	1,579,285	2.8	678,868	700,068	3.12
B, B'	28,457	49,625	74	1,213,825	1,213,825	0.0	543,838	543,838	0.00
C, C'	44,400	87,907	98	1,486,964	1,486,964	0.0	371,741	371,741	0.00
D, D'	22,968	29,157	27	1,400,515	1,411,490	0.8	563,183	616,170	9.41
E, E'	26,136	35,495	36	1,524,649	1,529,231	0.3	381,162	382,308	0.30
F, F'	42,457	84,119	98	1,416,260	1,416,260	0.0	384,700	384,700	0.00
G, G'	25,568	35,778	40	1,450,775	1,475,800	1.7	400,425	390,250	-2.54
APC _i			57			0.8			1.5

sitive to change. The largest average percent change (APC_i) occurred in the risk-related objective indicating that risk is the most sensitive objective. Regarding the effect of the quadratic risk scale, we illustrate in Figure 15 the optimal solution when $(w_1, w_2, w_3) = (1, 0, 0)$ and note, compared to the solution depicted in Figure 12, the rerouting of the flow of supplies to reduce transit through the high threat areas.

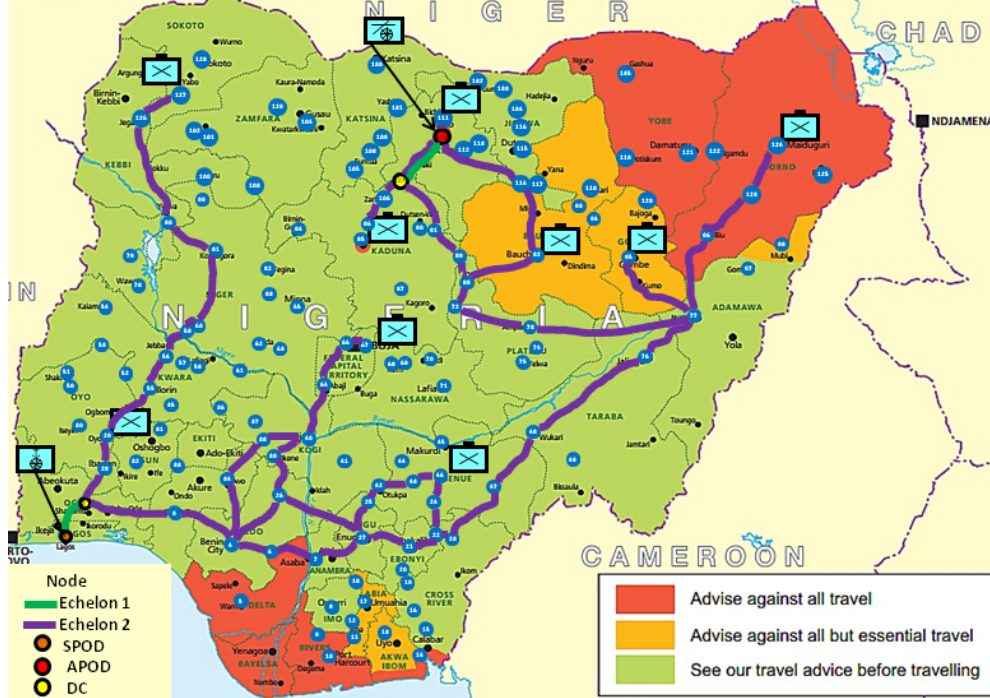


Figure 15. Optimal Solution for Instance A with Quadratic Risk Quantification [39]

The distance- and balance-related objectives experienced 0.8% and 1.5% difference between the linear and quadratic models, respectively. This result further supports

the identification of the insensitivity of the distance-based and workload-balancing objectives to the alternative quantification of risk.

V. Conclusion

Background

“Logistics ... as vital to military success as daily food is to daily work.”

Capt Alfred Thayer Mahan, USN [25]

This chapter provides the research conclusion from combining our developed model and a representative scenario. First, in Section 5.1 we provide a summary of the problem statement and methodology to solve the problem. Next, in Section 5.2 we discuss possible further research.

5.1 Conclusions

This work examined the problem of designing a military theater distribution network, wherein a decision maker selects Air and Sea Ports of Debarkation and intermediate logistical distribution centers, through which we route military supplies over a directed transportation network to meet demands. Considering alternative objectives to minimize the total risk encountered by transporting supplies, the total distance traveled by supplies, and the maximum per capita workload supported by transportation assets at a given echelon (i.e., port-to-distribution center versus distribution center to demands), we set forth a multi-objective, mixed-integer linear programming (MILP) formulation to solve this problem. For a given instance, we developed an automated tool that generates the MILP formulation and solves it using the commercial solver CPLEX. To demonstrate the efficacy of the model, we applied it to a representative instance and generated seven nondominated solutions on the efficient Pareto frontier. Computational tests indicated the relative variation of the three objectives among the instances examined and identified two of the objectives as being in tension for

the given scenario. Moreover, a successive analysis indicated that the distance-based and load-balancing objectives were relatively insensitive to an alternative metric for quantifying risk.

5.2 Future Research

In Section 1.3 we outlined four objectives for this research. The first research objective was met by developing an MILP for the movement of supplies through a transportation network. The second research objective was met by encoding in Visual Basic a program to construct the formulation within Microsoft Excel and invoke a commercial solver CPLEX (Version 12.5) to solve the Problem MTDNDP. The third research objective was met by using a scenario-based case study to demonstrate the model. The fourth research objective discussed examining extensions to the analysis. The remainder of this section discusses several areas for model improvement and areas for further research.

There are at least four areas for model improvement. First, an upper bound should be imposed on the total distance-based workload. This will ensure that, if a decision maker seeks to balance the workload between echelons of sustainment transport, it does not come at the cost of irrationally increasing the total distances traveled. Second, overall testing could be improved by using actual node throughput restrictions at APODs, SPODs, and DCs based on the availability of MHE. The availability of MHE varies on the task organization and number of CSSBs and this research made assumptions about the task organization. Model users will determine assets available for mission requirements and have a refined understanding of MHE available. Third, improve the intuitive understanding of q_2 and q_3 by converting them from supply-

distances and supply-distances per truck company to distance per unit of supply and average distance per truck company, respectively. Fourth, the model and solution methodology should be applied to additional scenarios to further assess its efficacy.

One suggested area for further research is the consideration of the reverse supply chain for the evacuation of casualties and materials. In the context of steady-state operations casualties, whether due to hostile fire or non-combat injuries, will occur. While some local medical facilities will have the capacity to treat casualties, some casualties will require evacuation to receive higher levels of care. This model could be modified to support the movement of casualties within the country or out of the country. Assuming there is no intent for the United States Government to maintain a long term presence in a country, all material deployed into the country will have to be redeployed to its home station. In the context of a redeploying force, the model could be modified to support a reverse supply chain.

Another area worthy of further research is to allow direct flow of supplies from a located POD to a FOB which could be either through ground or air lines of communication. Reasons for allowing direct flow from a POD to a FOB include but are not limited to an operational consideration that necessitates a rapid flow of supplies into a FOB or a located POD closer (time and/or distance) to a set of FOBS than it is to a located DC. In the case of direct flow of supplies via air from a POD to a FOB two additional reasons are to forgo the risk of ground high threat areas and non-existent ground lines of communication. Changes to the model to accommodate direct flow of supplies from a located POD to a FOB would be minimal. A third echelon of flow, which describes direct flow from a located POD to a FOB, would be introduced. The model would have to allow Echelon 3 flow out of a POD and into a FOB. The amount

of Echelon 3 flow out of a POD would be bounded the summation of the demand of the FOBs that it are closer to the POD than a located DC. The amount of Echelon 3 flow into a FOB would be bounded by the demand that FOB requires. A derivative to the direct flow of supplies from a located POD to a FOB is adding additional arcs to the network to represent intra-theater flow between airports, or, as appropriate, transportation via railway.

We used the Army's Command and General Staff College's *Theater Sustainment Battle Book* and easily accessible maps with varying degrees of fidelity to determine arc parameters for the network. These arc parameters were valid for our proof of principle, however use of this model in the future should involve better network data for the country or region that is analyzed. Some network data would necessarily classify this analysis, however there is available open source data that would make the results valuable to a decision maker. Finally, a more general network flow model should be examined, representing the possible loss of supplies either along arcs or at nodes due to pilferage.

Appendix A. Notional Nigeria Tactical Scenario

This map provides an overview of the Nigerian international borders, coast line, major cities, and major roads. The road network is the primary method for transporting supplies from A/SPODs to FOBs. The country of Nigeria in West Africa is the most populace African nation and contains an abundance of some of the worlds riches. Its neighbors are the Republic of Benin in the west, Chad and Cameroon in the east, and Niger in the north. Its coast in the south lies on the Gulf of Guinea in the Atlantic Ocean which gives it access to extensive trade opportunities.



UNCLASSIFIED MODELING SCENARIO

CJTF-Relief CONOPS



The AFIT of Today is the Air Force of Tomorrow.



Air University: The Intellectual and Leadership Center of the Air Force
Aim High...Fly - Fight - Win

The description below gives a summary of Nigeria's neighbors. Nigeria's stability is affected by events in neighboring countries. Nigerias neighbors vary in levels of economic, social, and political stability. As a nation formed by colonial powers in the 18th century, Nigerias borders are at times non-existent to tribal communities that extend beyond borders into neighboring countries. Along the border regions, and often times in the interior, loyalty to tribe supersedes loyalty to the Nigerian government.



UNCLASSIFIED MODELING SCENARIO
Regional Overview
Neighboring Countries



The AFIT of Today is the Air Force of Tomorrow.

Benin

Gained independence from France on 1 August 1960. Relatively stable republic government. Has three branches of government with political power resting with a prime minister. A majority of the people live along the southern coastline. Half of the population is split between Roman Catholic and Muslim. The economy is dependent on subsistence agriculture, cotton production, and regional trade.

Niger

Gained independence from France on 1 August 1960. Since independence the government was experienced numerous coups. A presidential election was held in 2011 with no clear winner. Consequently, the current president rules with little authority. More than 80% of the country is covered by the Sahara desert (central and north) and most people live in the south or west of the country. The population is mostly 80% Muslim. It is a landlocked nation whose economy is concentrated around subsistence farming. It is a developing country and consistently ranks low on the United Nations' Human Development Index.

Chad

Gained independence from France on 11 August 1960 and endured three decades of civil war and an invasion by Libya before peace was restored in 1990. By the constitution the president is elected every 5 years and contains almost all power. More than 54% of the country is Muslim and 34% Christian. By some indexes it is considered a failed state and consistently ranks as one of the poorest countries on the planet. Although there are oil reserves in the country, due to corruption over 80% of the country relies on subsistence farming.

Cameroon

Gained independence from France on 1 January 1960. It is considered a dominant-party president republic with an authoritarian president selected every seven years and enjoys social and political stability. Cameroon has developed roads, railways, agriculture systems, and large petroleum and timber industries. Indigenous and Christian beliefs each make up 40% of the population with the remaining 20% being Muslim. Agriculture and natural resources dominate the economy and Cameroon is an exporter of its goods to industrialized nations. Still, 30% of the population is unemployed and one-third of the population lives below international poverty standards.

Air University: The Intellectual and Leadership Center of the Air Force

Aim High...Fly - Fight - Win

3

The description below describes Nigeria's geography and topography in a military context. Nigeria is about twice the size of California and varies in terrain from tropical jungle in the south to a dry high plateau in the north. Lines of communication throughout Nigeria are marginal and even poor in rural areas.



UNCLASSIFIED MODELING SCENARIO

Regional Overview

Terrain



The AFIT of Today is the Air Force of Tomorrow.

- Overview
 - The total area is about twice the size of California and is the world's 32nd largest country. Nigeria has five major geographic regions: a low coastal zone along the Gulf of Guinea; hills and low plateaus north of the coastal zone; the Niger-Benue river valley; a broad stepped plateau stretching to the northern border with elevations exceeding 4000 feet; and a mountainous zone along the eastern border, which includes the country's highest point, Chappal Waddi (8000 feet). The Niger River (from west) and Benue River (from east) meet in the center of Nigeria and flow southward into the Gulf of Guinea.
- Observation and Fields of Fire:
 - Limited in the rain forests of the south and in all urban areas (Lagos - 11M, Abuja - 2M, and Kano - 3.5M)
 - In the central savannah and northern highlands there is less vegetation and visible distances are measured in miles.
- Obstacles:
 - In the south vehicular traffic is limited to roads due to rain forest and mountains.
 - In the central and northern sections of the country off-road is possible. In the central area the convergence of the Niger and Benue Rivers restrict some off-road movement. In the north there are steep valleys that can restrict some cross country movement.
- Cover and Concealment:
 - Excellent in the south. Entire villages and small settlements that are not documented by the government are plentiful.
 - The heavily vegetated areas of the central savannah along the major river provide excellent hiding locations.
 - The northern areas offer little vegetation, however wadis and broken/uneven terrain are plentiful.
- Avenues of Approach:
 - Almost 67,000 miles of surfaced road exist, but rarely maintained and can be treacherous at high speeds. All major cities are connected with some 4-lane highways. The railway system runs mostly north-south and is in poor repair. There are two major airports and 5 international airports.

15 August 2014


Air University: The Intellectual and Leadership Center of the Air Force

Aim High...Fly - Fight - Win

Analyst Workshop CS Study


4

The description below describes Nigerias human terrain in a military context. Nigeria is split along two lines; the northern dominated Muslim population and the southern dominated Christian population. There are four major tribes that align with geographic and religious lines. Competition for land and other natural resources between these groups has a history of violence, including a Civil War in 1967.




UNCLASSIFIED MODELING SCENARIO

AO OE & PMESII + PT



The AFIT of Today is the Air Force of Tomorrow.



OE:

- Militias are hidden within the population but are comprised of local leadership.
- Militias are dispersed but mainly in the northern region.
- To survive a majority of the population has chosen sides.

Political	Split along tribal, regional, and religious lines. Muslim north are from two tribes while the south has three tribes. Southern Nigerians have neglected northern concerns.
Military	Northerners see the military as heavy handed against Muslims and do not support it.
Economic	Revenue from rich natural resources in the south have been squandered due to corruption.
Social	Hausa and Fulani in the north. Yoruba in the south-west. Ijaws in the south and Igbo in the south-east. Nigerian Civil War (1967 – 1970) brought secessionist Igbo's back into Nigeria. Over 1M killed.
Infrastructure	Numerous road networks in poor condition due to neglect. Rail system in marginal condition and runs north to south.
Information	Muslim militias (predominantly Boko Haram) in the north benefit from world-wide Muslim support. Some foreign fighters have been found in Nigeria.
Physical Terrain	Complex terrain in the south limits ISR capabilities. Central and northern regions are more open.
Time	Muslim militias know that the government will not curb violence by Christian militias. Time is on the side of the Boko Haram as southern patience is waning.

15 August 2014

Air University: The Intellectual and Leadership Center of the Air Force
Aim High...Fly - Fight - Win

Analyst Workshop CS Study 5

The description below provides a historical accounting of how US forces became involved in the Nigerian conflict. The deployment of US forces to Nigeria is traced directly to unrest starting in 2014. The roots of this unrest dates back decades to Nigerias independence (1960) and even centuries to the colonial period (1800-1960).



UNCLASSIFIED MODELING SCENARIO

Road to War



The AFIT of Today is the Air Force of Tomorrow.

- 1960:** Gained independence from the United Kingdom. Underlying tensions between south (Christian, educated, more affluent, natural resource rich) and north (Muslim, less educated, few natural resources).
- 1999:** After 16 years of military rule a new constitution and civilian government put in place. Brutal military dictatorships were harsh on the north.
- 2002:** Militant Islamic group Boko Haram founded by Mohammed Yusuf.
- 2003:** The first violent Boko Haram attacks were against police stations.
- 2004:** More attacks against government. Hundreds of Boko Haram members rounded up and jailed. Christian militias kill over 600 Muslims in north.
- 2004 – 2008:** Boko Haram limits their attacks and builds strength.
- 2009 – 2010:** Boko Haram is able to attack prisons and free over 1000 jailed members. Conduct significant attacks all across the north. Christian militias organize in the north.
- 2011:** Boko Haram kill over 200 people and begin to move their attacks southward.
- 2012:** Over 500 deaths attributed to Boko Haram mostly in the north and central areas. President Goodluck Jonathan calls for talks with Boko Haram which quickly fall apart.
- 2013:** Boko Haram limits their attacks and builds strength. Kidnappings are rampant.
- 2014:** To date over 1200 people have been killed in Boko Haram attacks. The two most notable were the attacks on Bamboru and Ngala which left over 300 dead and the abduction of over 300 young girls from Chibok. No data on Christian caused deaths.
- 2014:** Thousands of Nigerians in the north and central have fled their homes and headed to neighboring countries or to the south. Reception in the south has been met with violence as overcrowding has increased tensions.

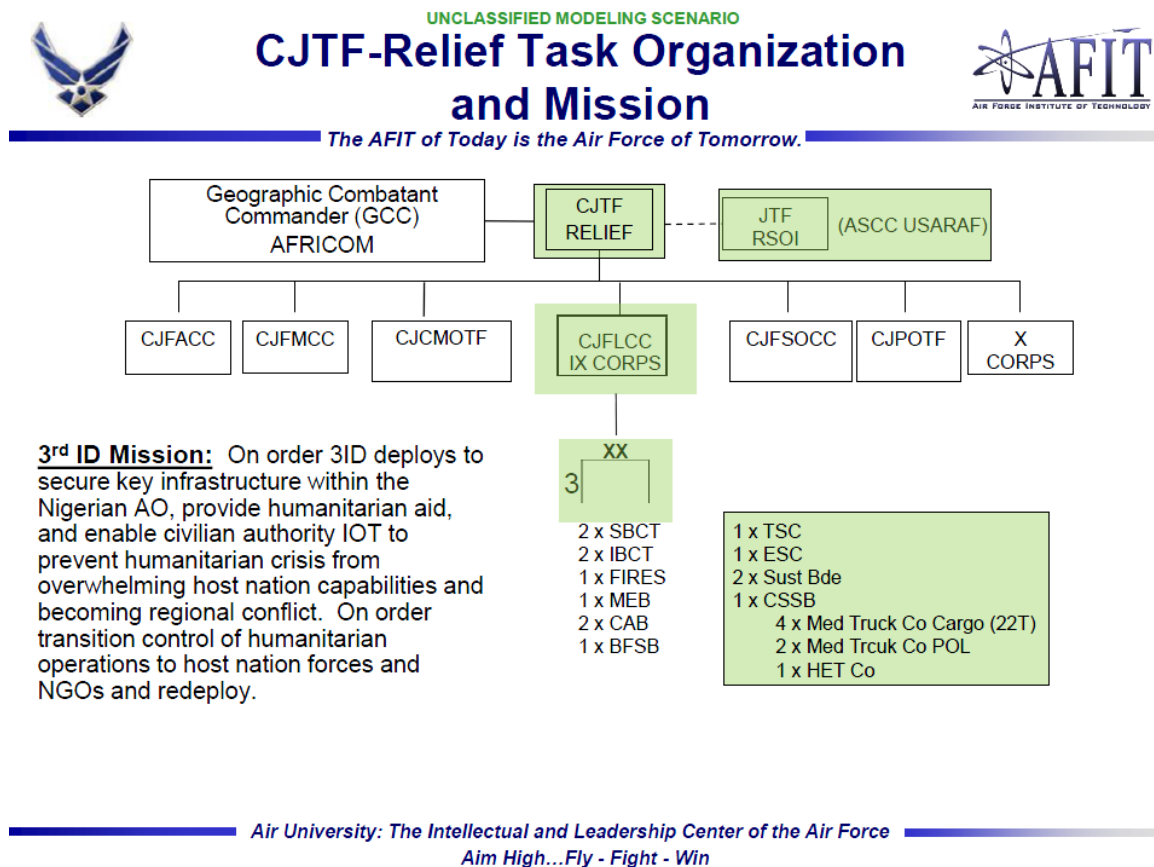
Air University: The Intellectual and Leadership Center of the Air Force

15 August 2014

Aim High...Fly - Fight - Win

Analyst Workshop CS Study 6

The description below displays all US military forces that are used in the operation. The wire diagram further shows the US Army forces that participate in the operation. US military operations in Nigeria are conducted under a JTF. Although the operation is conducted by US combat units, the mission is a humanitarian assistance mission.



The description below provides a general timeline and description for US military operations in Nigeria. The times are for planning purposes only and may vary depending on operational considerations on the ground. The six phase operation is planned for at least 180 days.



UNCLASSIFIED MODELING SCENARIO

Scenario Operational Timeline

CJTF Relief Phases of Operation



The AFIT of Today is the Air Force of Tomorrow.

Phase 0 - Shape political and social conditions in Nigeria	Phase I - Deter further dispersion and unrest with civilian population.	Phase II - Seize the initiative through rapid deployment of forces.	Phase III - Dominate the countryside to provide aid and maintain stability.	Phase IV - Stabilize unsettled regions along with host nation government forces and agencies.	Phase V - Enable civil authority to operate independent of coalition forces.
now to E-Day	E+1 to D-1	D-Day to D+60	D+61 to D+180	D+181 to D+365	D+ 180 - UTC

Phase 0: Strategic and political posturing by senior leaders.

Phase I: Military forces are alerted for pending deployment. Mission orders are issued and deliberate planning begins.

Phase II: Military forces deploy and develop the infrastructure for follow on forces. Coordinate with coalition partners and host nation for support. Protect the force.

Phase III: Conduct Humanitarian operation. Protect the force.

Phase IV: Enable host nation forces to conduct humanitarian operations

Phase V: Oversight of humanitarian operations and re-deploy.

Air University: The Intellectual and Leadership Center of the Air Force

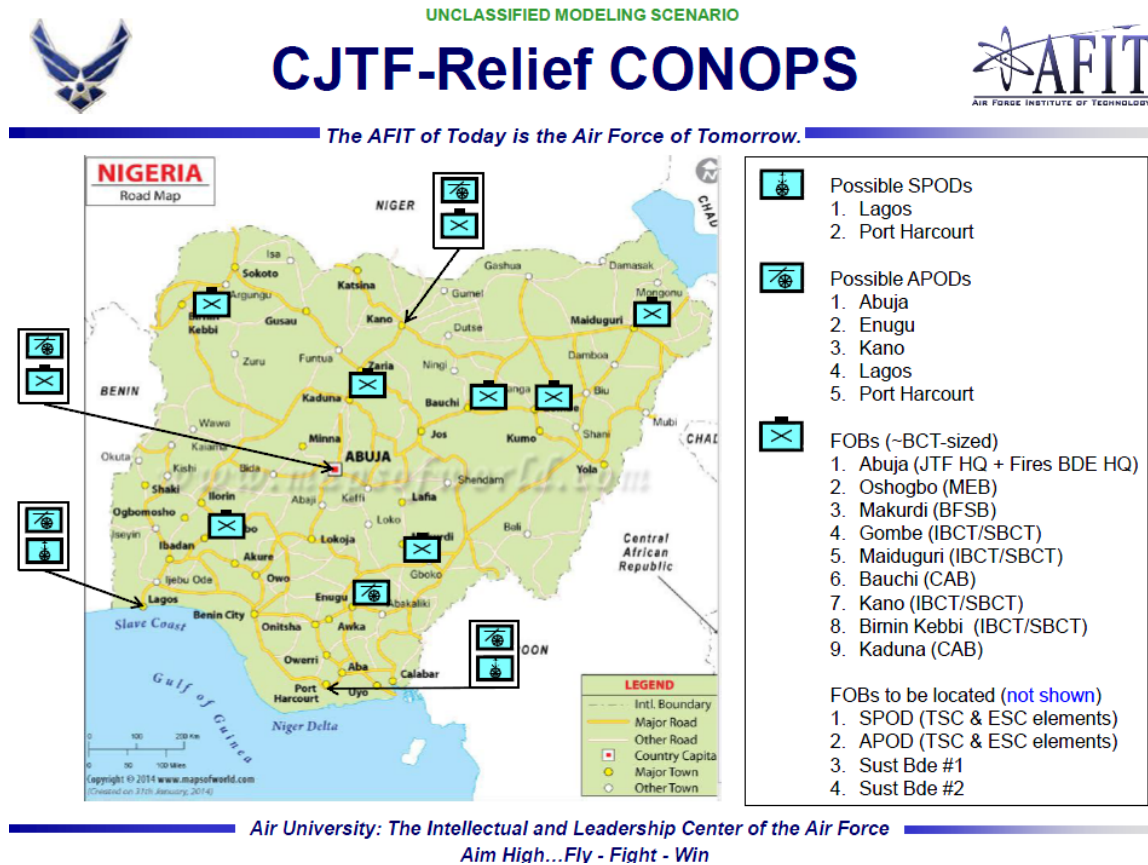
10 September 2014

Aim High...Fly - Fight - Win

Analyst Workshop CS Study

8

The map below provides an overview of the nine BCTs used to conduct the humanitarian mission. It also shows the location for possible A/SPODs.



Appendix B. VBA Code

The following screen shots of the VBA code are provided to document the code used to run the model developed in this research. They are not intended for the reader to use to reproduce the code as a copy of the code was provided to the sponsor; the copy in this annex is provided as a back-up.

The format for displaying the code is based on the five modules: master, node_data, arc_data, constraint_data, and cplexvba. Each screen shot is labeled at the top with the appropriate module name. Additionally, each module name is highlighted in blue in the "Project-VBAProject" window in the upper left side in each of the screen shots.

master (1 of 1)

The screenshot shows the VBA editor for a project named 'VBAPProject (20150202_input data)'. The Project Explorer on the left lists various sheets and modules. The 'master' module is selected, and its code is displayed in the main window. The code defines several public variables and a 'master' subroutine.

```

Public numNodes As Integer
Public lastRow As Long
Public lastRow2 As Long
Public lastCol As Long
Public lastCol2 As Long
Public firstRow As Long
Public numArcs As Integer
Public dijRange As Range
Public cij1Range As Range
Public xij1Range As Range
Public uijRange As Range
Public cij2Range As Range
Public xij2Range As Range
Public const47 As Integer
Public const48 As Integer
Public const49 As Integer
Public const50 As Integer

Sub master()

    ' Stops the screen from updating i.e. will speed up code.
    'Application.ScreenUpdating = False

    ' Clears the worksheet below Row 11
    Rows("11:65536").Delete

    ' Clears the OF, DV, and constraints out of CPLEX
    CPXclear

    ' Centers the entire worksheet
    Cells.Select
    With Selection
        .HorizontalAlignment = xlCenter
        .VerticalAlignment = xlBottom
        .WrapText = False
        .Orientation = 0
        .AddIndent = False
        .IndentLevel = 0
        .ShrinkToFit = False
        .ReadingOrder = xlContext
        .MergeCells = False
    End With

    Call nodeData

    Call arcData

    Call constraints

    MsgBox "Data is now uploaded."

    CPXsolve

    ' Allows the screen to update.
    Application.ScreenUpdating = True

End Sub

```

node_data (1 of 5)

The screenshot shows the VBA editor for a project named 'VBAPROJECT (20150202_input data)'. The Project Explorer on the left lists several sheets (Sheet1 through Sheet17) and modules (arc_data, constraint_data, cplexvba, master, and node_data). The Properties window at the bottom left shows the 'node_data' module. The main window displays the VBA code for the 'node_data' module, which includes variable declarations, formatting for rows 11 and 12, and writing column headings for 'Node-Specific Data'.

```

Option Explicit
Sub nodeData()

Dim myFile As String
Dim nodeData As String
Dim lastComma As Integer
Dim i As Integer
Dim j As Integer
Dim q As Integer
Dim row_number As Integer
Dim k As Variant
Dim eachitem As Variant
Dim nodeCount As Integer

' Makes the gap between 'Individual Values' and 'node-specific data' YELLOW.
Rows("11:11").Select
    With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 65535
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With

' Makes the 'Node-specific data' row RED.
Rows("12:12").Select
    With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .Color = 255
        .TintAndShade = 0
        .PatternTintAndShade = 0
    End With

' Writes the column heading "Node-Specific Data".
Sheets("Sheet1").Select
Range("b12").Select
ActiveCell.FormulaR1C1 = "Node-Specific Data"

' Writes the 1st column heading "Node-Specific Data".
Sheets("Sheet1").Select
Range("A13").Select
ActiveCell.FormulaR1C1 = "Node"
Range("b13").Select
ActiveCell.FormulaR1C1 = "b_i"
Range("c13").Select
ActiveCell.FormulaR1C1 = "U_i^a"
Range("d13").Select
ActiveCell.FormulaR1C1 = "U_i^s"
Range("e13").Select
ActiveCell.FormulaR1C1 = "U_i^dc"

' Shades the column heading for "Node_Specific Data" of first matrix
Range("a13:e13").Select
    With Selection.Interior
        .Pattern = xlSolid
        .PatternColorIndex = xlAutomatic
        .ThemeColor = xlThemeColorDark1
        .TintAndShade = -0.349986266670736
        .PatternTintAndShade = 0
    End With

```


node_data (2 of 5)

The screenshot shows the VBA editor for a project named 'VBAProject (20150202_input data)'. The Project Explorer on the left lists several sheets (Sheet1 through Sheet17) and a 'Modules' folder containing 'arc_data', 'constraint_data', 'cplexvba', 'master', and 'node_data'. The 'node_data' module is selected, and its properties are shown in the bottom-left window. The main editor area displays the following VBA code:

```

' Writes the 1st column heading "Node-Specific Data".
Sheets("Sheet1").Select
Range("g13").Select
ActiveCell.FormulaR1C1 = "y_i^a"
Range("h13").Select
ActiveCell.FormulaR1C1 = "y_i^s"
Range("i13").Select
ActiveCell.FormulaR1C1 = "y_i^dc"
Range("j13").Select
ActiveCell.FormulaR1C1 = "z_i^c"
Range("k13").Select
ActiveCell.FormulaR1C1 = "z_i^nc"

' Shades the column heading for the decision variable (second) matrix.
Range("g13:k13").Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Places title "Decision Variable Matrix" above the decision variable matrix.
Range("i12").Select
ActiveCell.FormulaR1C1 = "Decision Variable Matrix"

' Reads the node data from a .txt file
myFile = "I:\My Documents\Thesis\Chapter 4\VBA Interface\node_data1.txt"
myFile = "I:\My Documents\Thesis\Chapter 4\VBA Interface\node_data2.txt"
Open myFile For Input As #1

' Converts the data in the .txt file into Excel-cell format
nodeCount = 0
Cells(1, 1).Select
row_number = 13
Do Until EOF(1)

    Line Input #1, k
    eachitem = Split(k, ",")

    ActiveCell.Offset(row_number, 0).Value = eachitem(0)
    ActiveCell.Offset(row_number, 1).Value = eachitem(1)
    ActiveCell.Offset(row_number, 2).Value = eachitem(2)
    ActiveCell.Offset(row_number, 3).Value = eachitem(3)
    ActiveCell.Offset(row_number, 4).Value = eachitem(4)
    row_number = row_number + 1
    nodeCount = nodeCount + 1

Loop

' Finds the position of the last row
lastRow = ActiveSheet.Cells(Rows.Count, "a").End(xlUp).Row

Close #1
' Assigns the number of nodes to 'numNodes'
numNodes = nodeCount

```

node_data (3 of 5)

The screenshot displays the VBA Project window for 'VBAPProject (20150202_input data)'. The left pane shows the 'Modules' list with 'node_data' selected. The right pane shows the code for the 'node_data' module, which is currently in the 'General' tab. The code is as follows:

```

' Makes a grid system for the 'Node-Specific Data' matrix
Range("A13:E" & lastRow).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Makes a grid system for the 'Decision Variable Matrix'
Range("G13:k" & lastRow).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Places "0" in all of the cells of the "Decision Variable Matrix"
Range(Cells(14, 7), Cells(14 + numNodes - 1, 11)) = 0

' Shades the "Node" row for "Node-Specific Data" of first matrix
Range("a13:a" & lastRow).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Shades the APD decision variables (y_i^a) blue that are possible solutions in the DV matrix
For i = lastRow - numNodes + 1 To lastRow
    If Cells(i, 3) = 0 Then
        Else
            CPXAddVariable Variable:=Cells(i, 7), Lb:=0, Binary:=True 'Add CPLEX decision variables everytime cell turns blue
            Cells(i, 7).Select
            With Selection.Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .Color = 15773696
                .TintAndShade = 0
                .PatternTintAndShade = 0
            End With
        End If
    End If
Next i

```

The Properties window at the bottom left shows the 'node_data' module selected, with the 'Name' property set to 'node_data'.

node_data (4 of 5)

The screenshot shows the VBA editor for a project named 'VBAPProject'. The Project Explorer on the left lists the 'Microsoft Excel Objects' (Sheets1 through Sheet17) and 'Modules' (arc_data, constraint_data, cplexvba, master, and node_data). The Properties window at the bottom left shows the 'node_data' module. The main editor window displays the following VBA code:

```

' Shades the AFOD decision variables (y_i^a) blue that are possible solutions in the DV matrix
For i = lastRow - numNodes + 1 To lastRow
    If Cells(1, 3) = 0 Then
        Else
            CPXaddVariable Variable:=Cells(1, 7), Lb:=0, Binary:=True 'Add CPLEX decision variables everytime cell turns blue
            Cells(1, 7).Select
            With Selection.Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .Color = 15773696
                .TintAndShade = 0
                .PatternTintAndShade = 0
            End With
        End If
    Next i
Next i

' Shades the SPOD decision variables (y_i^s) blue that are possible solutions in the DV matrix
For i = lastRow - numNodes + 1 To lastRow
    If Cells(1, 4) = 0 Then
        Else
            Cells(1, 8).Select
            CPXaddVariable Variable:=Cells(1, 8), Lb:=0, Binary:=True 'Add CPLEX decision variables everytime cell turns blue
            With Selection.Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .Color = 15773696
                .TintAndShade = 0
                .PatternTintAndShade = 0
            End With
        End If
    Next i

' Shades the DC decision variables (y_i^dc) blue that are possible solutions in the DV matrix
For i = lastRow - numNodes + 1 To lastRow
    If Cells(1, 5) = 0 Then
        Else
            CPXaddVariable Variable:=Cells(1, 9), Lb:=0, Binary:=True 'Add CPLEX decision variables everytime cell turns blue
            Cells(1, 9).Select
            With Selection.Interior
                .Pattern = xlSolid
                .PatternColorIndex = xlAutomatic
                .Color = 15773696
                .TintAndShade = 0
                .PatternTintAndShade = 0
            End With
        End If
    Next i

```

node_data (5 of 5)

The screenshot displays the VBA Project window for 'VBAPProject'. The left pane shows the project structure, including 'Microsoft Excel Objects' (Sheets 1 through 17) and 'Modules' (arc_data, constraint_data, cplexvba, master, and node_data). The 'node_data' module is selected. The right pane shows the VBA code for the 'node_data' module, which includes logic for shading decision variables and adding them to the CPLEX model.

```

With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .Color = 15773696
    .TintAndShade = 0
    .PatternTintAndShade = 0
End With
End If
Next i

' Shades the DC decision variables (y_i^dc) blue that are possible solutions in the DV matrix
' For i = lastRow - numNodes + 1 To lastRow
'     If Cells(i, 5) = 0 Then
'     Else
'         CPXaddVariable Variable:=Cells(i, 9), Lb:=0, Binary:=True 'Add CPLEX decision variables everytime cell turns blue
'         Cells(i, 9).Select
'
'         With Selection.Interior
'             .Pattern = xlSolid
'             .PatternColorIndex = xlAutomatic
'             .Color = 15773696
'             .TintAndShade = 0
'             .PatternTintAndShade = 0
'         End With
'     End If
' End With
Next i

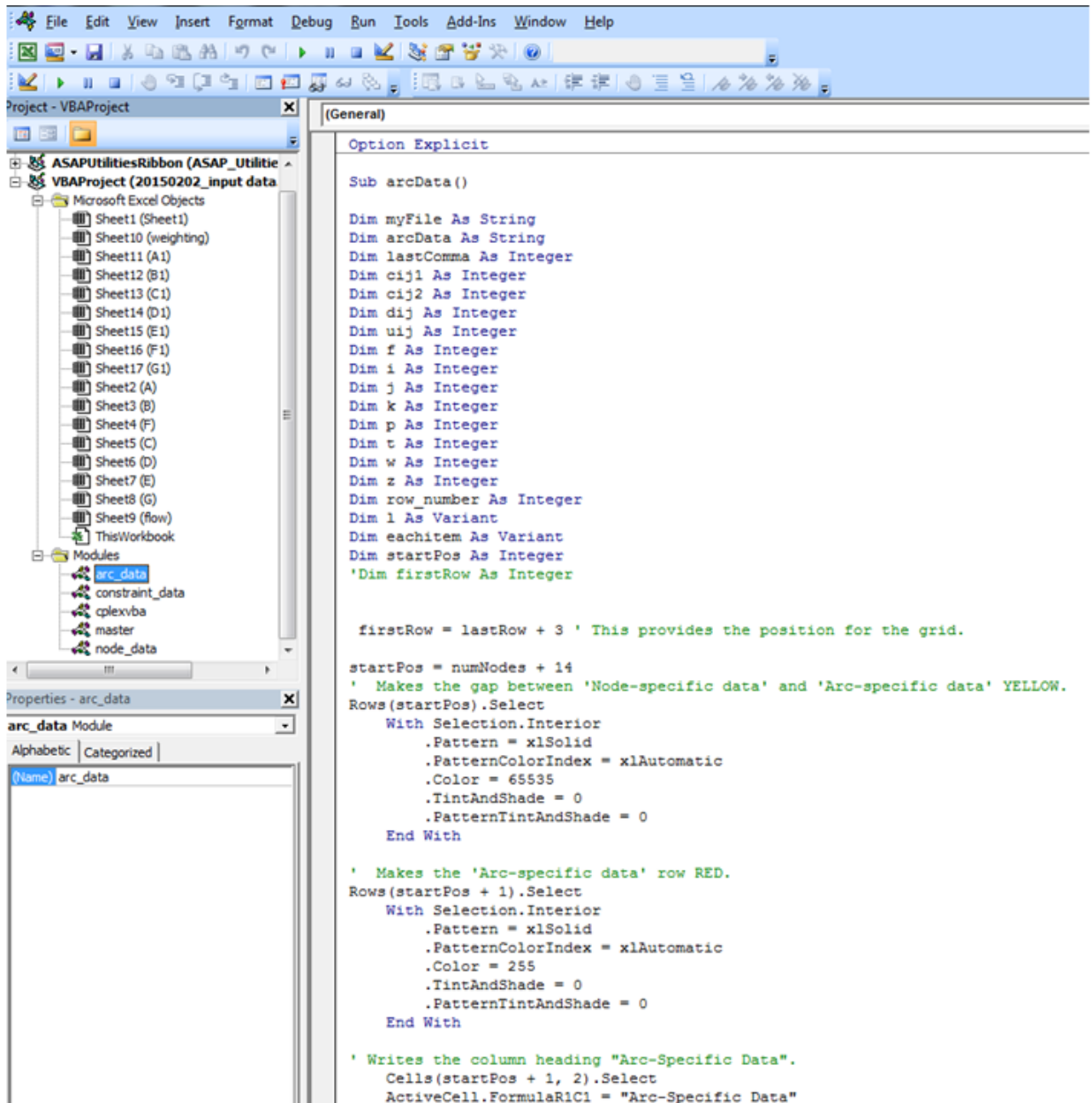
' This section shades DC decision variables (y_i^dc) blue that are possible solutions in the DV matrix _
and reads them into CPLEX as DVs.
Dim ydv As Range
Set ydv = Range(Cells(14, 9), Cells(14 + numNodes - 1, 9))
CPXaddVariable Variable:=ydv, Lb:=0, Binary:=True
ydv.Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .Color = 15773696
    .TintAndShade = 0
    .PatternTintAndShade = 0
End With

' This section shades DVs (z_i^c, z_i^nc) blue if they return a non-zero value and reads them into CPLEX as DVs.
Dim zdv As Range
Set zdv = Range("j14:k" & lastRow)
CPXaddVariable Variable:=zdv, Lb:=0

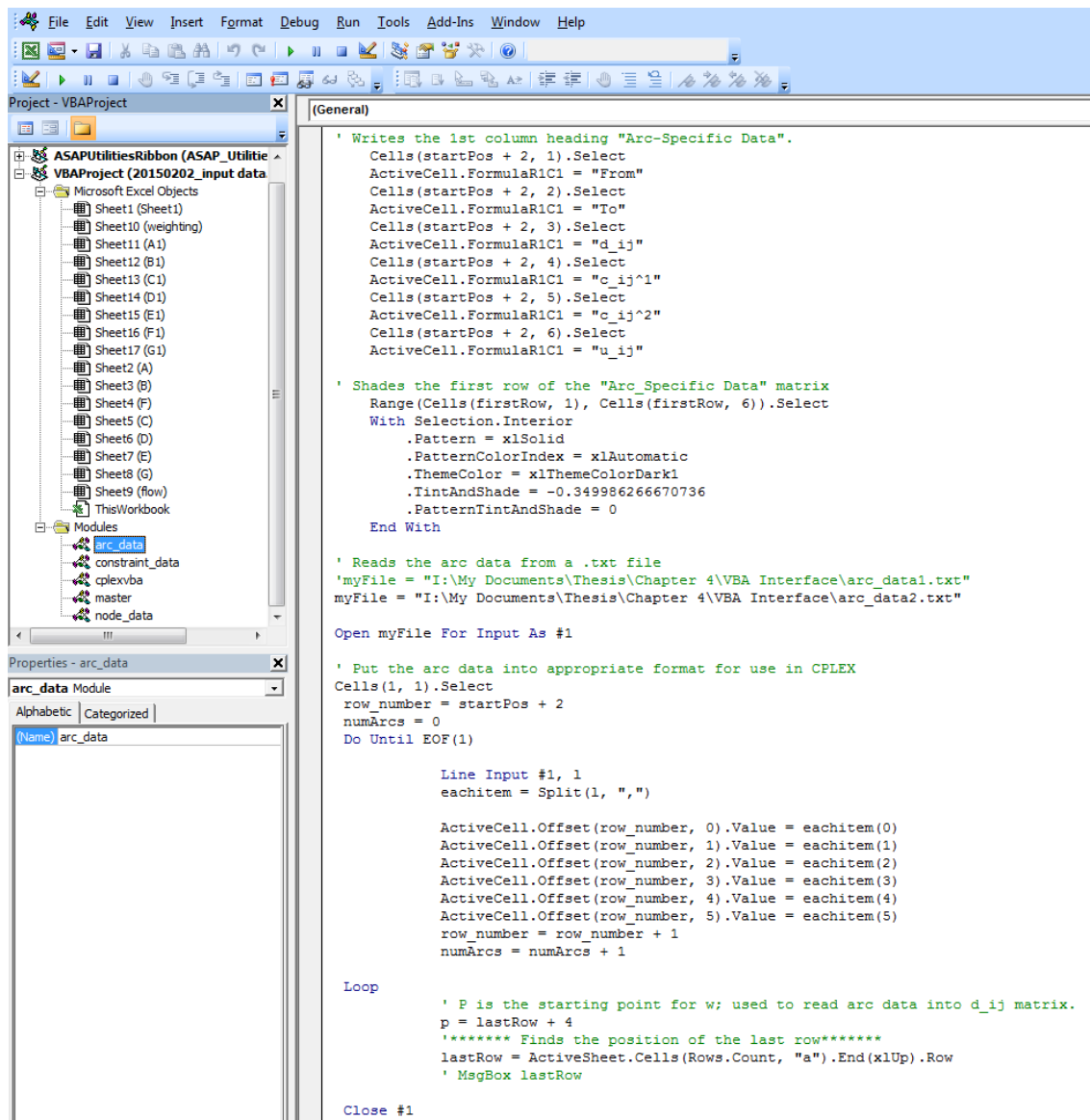
Range("j14:k" & lastRow).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .Color = 15773696
    .TintAndShade = 0
    .PatternTintAndShade = 0
End With
End Sub

```

arc_data (1 of 14)



arc_data (2 of 14)



arc_data (3 of 14)

The screenshot displays the VBA editor for a project named 'VBAPProject'. The 'Project - VBAPProject' window on the left shows a hierarchy of 'Microsoft Excel Objects' (Sheets 1 through 17) and 'Modules' (arc_data, constraint_data, cplexvba, master, node_data). The 'arc_data' module is selected, and its properties are shown in the 'Properties - arc_data' window. The main window displays the VBA code for the 'arc_data' module, which includes comments and code for creating a grid system, shading cells, and writing decision variable matrices.

```

' Makes a grid system for the Arc-specific datamatrix
Range(Cells(firstRow, 1), Cells(lastRow, 6)).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Shades the first column of the "Arc_Specific Data" matrix.
Range(Cells(firstRow, 1), Cells((firstRow + numArcs), 2)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Writes the column heading for the x_ij^1 and x_ij^2 decision variable matrix.
Cells(startPos + 2, 8).Select
ActiveCell.FormulaR1C1 = "x_ij^1"
Cells(startPos + 2, 9).Select
ActiveCell.FormulaR1C1 = "x_ij^2"

' Shades the heading for the x_ij^1 and x_ij^2 decision variable matrix.
Range(Cells(startPos + 2, 8), Cells(startPos + 2, 9)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Makes the grid system for the x_ij^1 and x_ij^2 decision variable matrix.
Range(Cells(startPos + 2, 8), Cells(lastRow, 9)).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

.....
' d_ij, u_ij, c_ij^1, c_ij^2, x_ij^1, and x_ij^2 matrix
.....

firstRow = lastRow + 4
'..... Begin d_ij matrix .....
' Places the d_ij heading
Cells(firstRow, 1).Select
ActiveCell.FormulaR1C1 = "d_ij"

```

arc_data (4 of 14)

The screenshot displays the VBA editor for a project named 'VBAPProject (20150202_input data)'. The 'Project - VBAPProject' window on the left shows the 'Modules' list with 'arc_data' selected. The 'Properties - arc_data' window below it shows the 'arc_data' module. The main editor window shows the VBA code for the 'arc_data' module, which includes comments and code for setting up the 'd_ij' matrix, shading cells, and applying format conditions.

```

' Places the row designation for d_ij matrix
j = 1
For i = (firstRow + 1) To (firstRow + numNodes)
    Cells(i, 2) = j
    j = j + 1
Next i

' Places the column designation for d_ij
k = 1
For i = 3 To numNodes + 2
    Cells(firstRow, i) = k
    k = k + 1
Next i

' Makes a grid system for the d_ij matrix
Range(Cells(firstRow, 2), Cells(firstRow + numNodes, numNodes + 2)).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Shades the first row of the "d_ij" matrix
Range(Cells(lastRow + 4, 2), Cells(lastRow + 4, numNodes + 2)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Shades the first column of the "d_ij" matrix
Range(Cells(lastRow + 4, 2), Cells(lastRow + 4 + numNodes, 2)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Places "0" in all of the cells of d_ij
Range(Cells(firstRow + 1, 3), Cells(firstRow + numNodes, numNodes + 2)) = 0

' Shades d_ij brown if a non-zero value
Range(Cells(firstRow + 1, 3), Cells(firstRow + numNodes, numNodes + 2)).Select
Selection.FormatConditions.Add Type:=xlCellValue, Operator:=xlNotEqual, _
    Formula1:="=0"
Selection.FormatConditions(Selection.FormatConditions.Count).SetFirstPriority
With Selection.FormatConditions(1).Interior
    .PatternColorIndex = xlAutomatic
    .Color = 3368601
    .TintAndShade = 0
End With
Selection.FormatConditions(1).StopIfTrue = False

```


arc_data (5 of 14)

```

' Places the d_ij's in the appropriate cells
For w = p To lastRow
    f = Cells(w, 1)
    t = Cells(w, 2)
    dij = Cells(w, 3)
    For i = (firstRow + 1) To (numNodes + firstRow) ' This checks all the row values.
        For j = 3 To (numNodes + 2) ' This checks all the column values.
            If (Cells(i, 2) = f And Cells(firstRow, j) = t) Then
                Cells(i, j) = dij
            End If
        Next j
    Next i
Next w

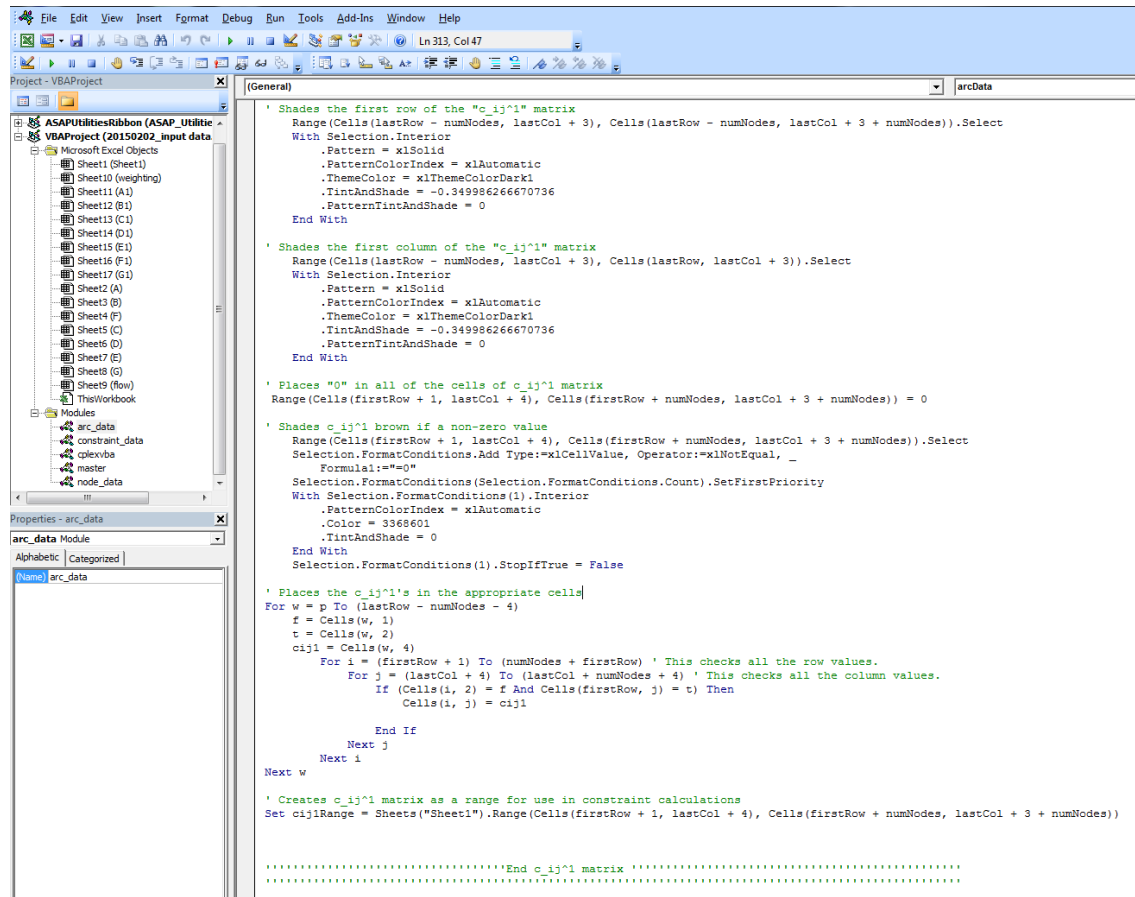
' Creates d_ij matrix as a range for use in constraint calculations
Set dijRange = Sheets("Sheet1").Range(Cells(firstRow + 1, 3), Cells(firstRow + numNodes, numNodes + 2))

' End d_ij matrix

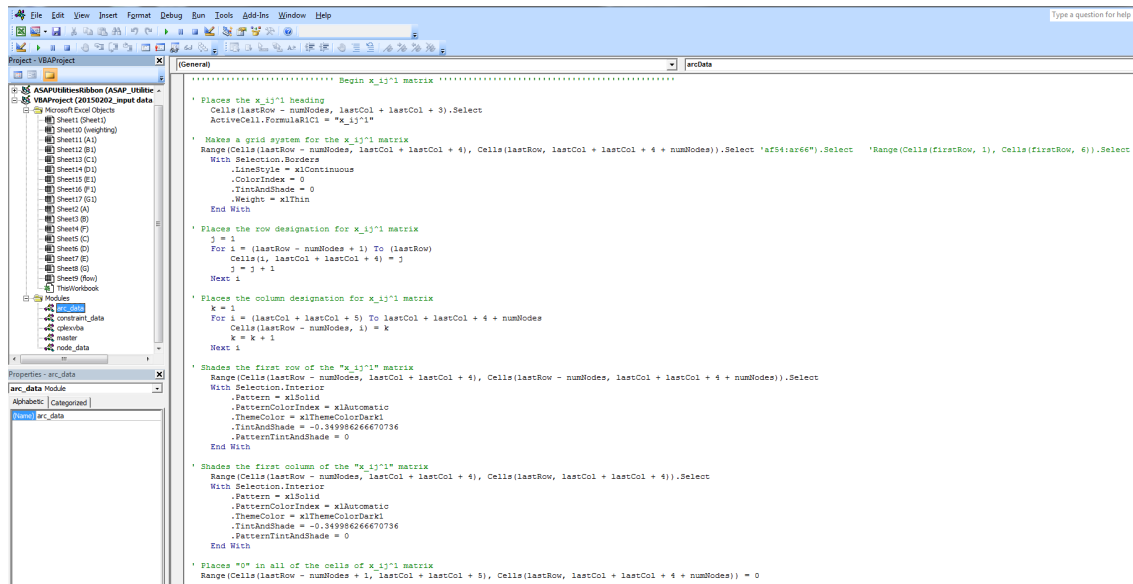
'Begin c_ij^1 matrix
' Finds the position of the last col
' Finds the position of the last column
lastRow = ActiveSheet.Cells(Rows.Count, "b").End(xlUp).Row
lastCol = Cells(lastRow, Columns.Count).End(xlToLeft).Column
' Places the d_ij heading
Cells(firstRow, lastCol + 2).Select
ActiveCell.FormulaR1C1 = "c_ij^1"
' Places the row designation for c_ij^1 matrix
j = 1
For i = (firstRow + 1) To (firstRow + numNodes)
    Cells(i, lastCol + 3) = j
    j = j + 1
Next i
' Places the column designation for c_ij^1 matrix
k = 1
For i = (lastCol + 4) To lastCol + 3 + numNodes
    Cells(firstRow, i) = k
    k = k + 1
Next i
' Makes a grid system for the c_ij^1 matrix
Range(Cells(firstRow, lastCol + 3), Cells(firstRow + numNodes, lastCol + 3 + numNodes)).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

```

arc_data (6 of 14)



arc_data (7 of 14)



arc_data (8 of 14)

```

' Turns the cells of the x_ij^1 matrix blue if it is a potential solution.
z = 1 ' The counter to color blue cells in the x_ij^1 and x_ij^2 matrices.
For w = p To (p + numArcs)
    f = Cells(w, 1) ' reads the "from" node from the "arc-specific data" matrix
    t = Cells(w, 2) ' reads the "to" node from the "arc-specific data" matrix
    For i = (firstRow + 1) To (numNodes + firstRow) ' This checks all the row values.
        For j = (2 * lastCol + 5) To (2 * lastCol + 4 + numNodes) ' This checks all the column values.
            If (Cells(i, (2 * lastCol + 4)) = f And Cells(firstRow, j) = t) Then
                Cells(i, j).Select ' Colors the cell blue
                With Selection.Interior
                    .PatternColorIndex = xlAutomatic
                    .Color = 3368601
                    .TintAndShade = 0
                End With
                Cells(firstRow - 4 - numArcs + z, 8).Select ' Colors the x_ij^1 cell in the "Arc-specific data" section blue
                With Selection.Interior
                    .PatternColorIndex = xlAutomatic
                    .Color = 15773696
                    .TintAndShade = 0
                End With
                'Cells(firstRow - 4 - numArcs + z, 8) = "=" & Cells(i, j).Address(ReferenceStyle:=xlA1, _
                'RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the x_ij^1 value into the into the Blue cell in the "Arc-specific Data Section".
                Cells(i, j) = "=" & Cells(firstRow - 4 - numArcs + z, 8).Address(ReferenceStyle:=xlA1, _
                RowAbsolute:=False, ColumnAbsolute:=False)
            End With
            z = z + 1
        Next j
    Next i
Next w

' The code below existed before my attempts to shade x_ij^1's blue if the u_ij > 0. I will use it if I can get BVA to call CPLEX.
' Shades x_ij^1 blue if it returns a non-zero Value
Range(Cells(lastRow - numNodes + 1, lastCol + lastCol + 5), Cells(lastRow, lastCol + lastCol + 4 + numNodes)).Select
Selection.FormatConditions.Add Type:=xlCellValue, Operator:=<=, Formula1:="=0"
Selection.FormatConditions(Selection.FormatConditions.Count).SetFirstPriority
With Selection.FormatConditions(1).Interior
    .PatternColorIndex = xlAutomatic
    .Color = 15773696
    .TintAndShade = 0
End With
Selection.FormatConditions(1).StopIfTrue = False

' Creates x_ij^1 matrix as a range for use in constraint calculations
Set xij1Range = Sheets("Sheet1").Range(Cells(lastRow - numNodes + 1, lastCol + lastCol + 5), Cells(lastRow, lastCol + lastCol + 4 + numNodes))

' End x_ij^1 matrix

```

arc_data (9 of 14)

```

.....'Begin u_ij matrix '.....
' Places the u_ij heading
firstRow = lastRow + 2
Cells(firstRow, 1).Select
ActiveCell.FormulaR1C1 = "u_ij"

' Places the row designation for u_ij matrix
j = 1
For i = (firstRow + 1) To (firstRow + numNodes)
    Cells(i, 2) = j
    j = j + 1
Next i

' Places the column designation for u_ij
k = 1
For i = 3 To numNodes + 2
    Cells(firstRow, i) = k
    k = k + 1
Next i

' Makes a grid system for the u_ij matrix
Range(Cells(firstRow, 2), Cells(firstRow + numNodes, numNodes + 2)).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Places "0" in all of the cells of u_ij
Range(Cells(firstRow + 1, 3), Cells(firstRow + numNodes, numNodes + 2)) = 0

' Shades c_ij^1 brown if a non-zero value
Range(Cells(firstRow + 1, 3), Cells(firstRow + numNodes, numNodes + 2)).Select
Selection.FormatConditions.Add Type:=xlCellValue, Operator:=xlNotEqual, _
    Formula1:="=0"
Selection.FormatConditions(Selection.FormatConditions.Count).SetFirstPriority
With Selection.FormatConditions(1).Interior
    .PatternColorIndex = xlAutomatic
    .Color = 3368601
    .TintAndShade = 0
End With
Selection.FormatConditions(1).StopIfTrue = False

' Shades the first row of the "u_ij" matrix
Range(Cells(firstRow, 2), Cells(firstRow, numNodes + 2)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Shades the first column of the "u_ij" matrix
Range(Cells(firstRow, 2), Cells(firstRow + numNodes, 2)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With
    
```

arc_data (10 of 14)

The screenshot displays the VBA editor for a project named 'VBAPProject'. The Project Explorer on the left shows the 'Modules' section with 'arc_data' selected. The Properties window at the bottom left shows the 'arc_data' module. The VBA code editor on the right contains the following code:

```

' Places the u_ij's in the appropriate cells
For w = p To (lastRow - numNodes - 4)
    f = Cells(w, 1)
    t = Cells(w, 2)
    uij = Cells(w, 6)
    For i = (firstRow + 1) To (numNodes + firstRow) ' This checks all the row values.
        For j = 3 To (numNodes + 2) ' This checks all the column values.
            If (Cells(i, 2) = f And Cells(firstRow, j) = t) Then
                Cells(i, j) = uij
            End If
        Next j
    Next i
Next w

' Creates u_ij matrix as a range for use in constraint calculations
Set uijRange = Sheets("Sheet1").Range(Cells(firstRow + 1, 3), Cells(firstRow + numNodes, numNodes + 2))
uijRange.Select

.....End u_ij matrix .....

.....Begin c_ij^2 matrix .....
' Places the c_ij^2 heading
Cells(firstRow, lastCol + 2).Select
ActiveCell.FormulaR1C1 = "c_ij^2"
' Places the row designation for c_ij^2 matrix
j = 1
For i = (firstRow + 1) To (firstRow + numNodes)
    Cells(i, lastCol + 3) = j
    j = j + 1
Next i
' Places the column designation for c_ij^2 matrix
k = 1
For i = (lastCol + 4) To lastCol + 3 + numNodes
    Cells(firstRow, i) = k
    k = k + 1
Next i
' Makes a grid system for the c_ij^2 matrix
Range(Cells(firstRow, lastCol + 3), Cells(firstRow + numNodes, lastCol + 3 + numNodes)).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Places "0" in all of the cells of c_ij^2 matrix
Range(Cells(firstRow + 1, lastCol + 4), Cells(firstRow + numNodes, lastCol + 3 + numNodes)) = 0

```

arc_data (11 of 14)

The screenshot displays the VBA editor for a project named 'VBAProject (20150202_input data)'. The 'Project - VBAProject' window on the left shows the 'Modules' list with 'arc_data' selected. The 'Properties - arc_data' window below it shows the 'arc_data Module' with the 'Name' property set to 'arc_data'. The main editor window shows the VBA code for the 'arc_data' module, which is organized into several sections: formatting the 'c_ij^2' matrix, shading the first row and column, and placing the matrix values in the appropriate cells.

```

' Shades c_ij^2 brown if a non-zero value
Range(Cells(firstRow + 1, lastCol + 4), Cells(firstRow + numNodes, lastCol + 3 + numNodes)).Select
Selection.FormatConditions.Add Type:=xlCellValue, Operator:=xlNotEqual, _
    Formula1:="=0"
Selection.FormatConditions(Selection.FormatConditions.Count).SetFirstPriority
With Selection.FormatConditions(1).Interior
    .PatternColorIndex = xlAutomatic
    .Color = 3368601
    .TintAndShade = 0
End With
Selection.FormatConditions(1).StopIfTrue = False

' Shades the first row of the "c_ij^2" matrix
Range(Cells(firstRow, lastCol + 3), Cells(firstRow, lastCol + 3 + numNodes)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Shades the first column of the "c_ij^2" matrix
Range(Cells(firstRow, lastCol + 3), Cells(firstRow + numNodes, lastCol + 3)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

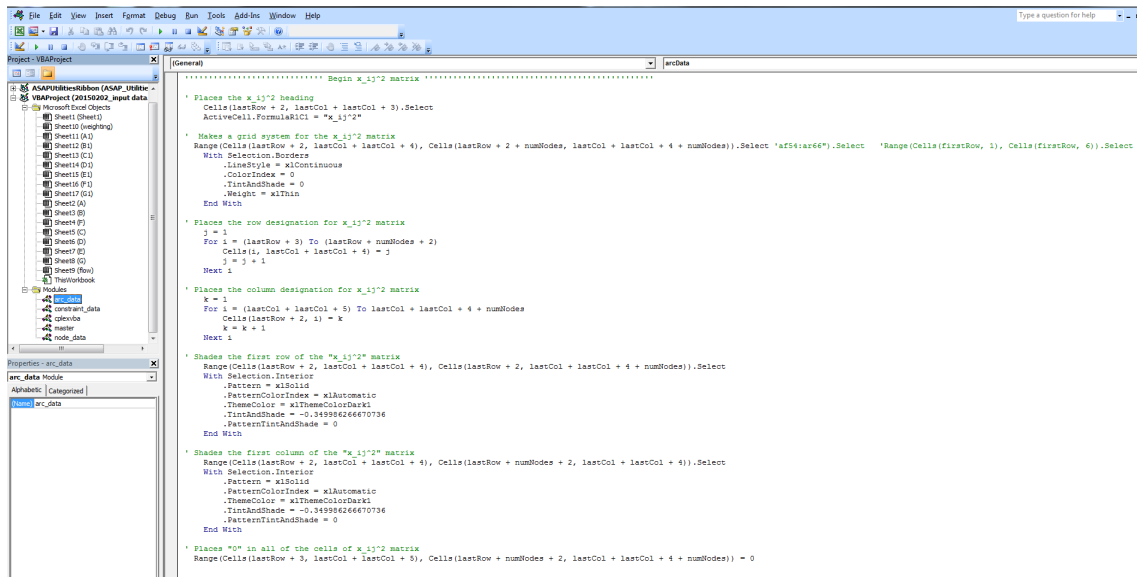
' Places the c_ij^2's in the appropriate cells
For w = p To (lastRow - numNodes - 4)
    f = Cells(w, 1)
    t = Cells(w, 2)
    cij2 = Cells(w, 5)
    For i = (firstRow + 1) To (numNodes + firstRow) ' This checks all the row values.
        For j = (lastCol + 4) To (lastCol + numNodes + 4) ' This checks all the column values.
            If (Cells(i, 2) = f And Cells(firstRow, j) = t) Then
                Cells(i, j) = cij2
            End If
        Next j
    Next i
Next w

' Creates c_ij^2 matrix as a range for use in constraint calculations
Set cij2Range = Sheets("Sheet1").Range(Cells(firstRow + 1, lastCol + 4), Cells(firstRow + numNodes, lastCol + 3 + numNodes))
cij2Range.Select

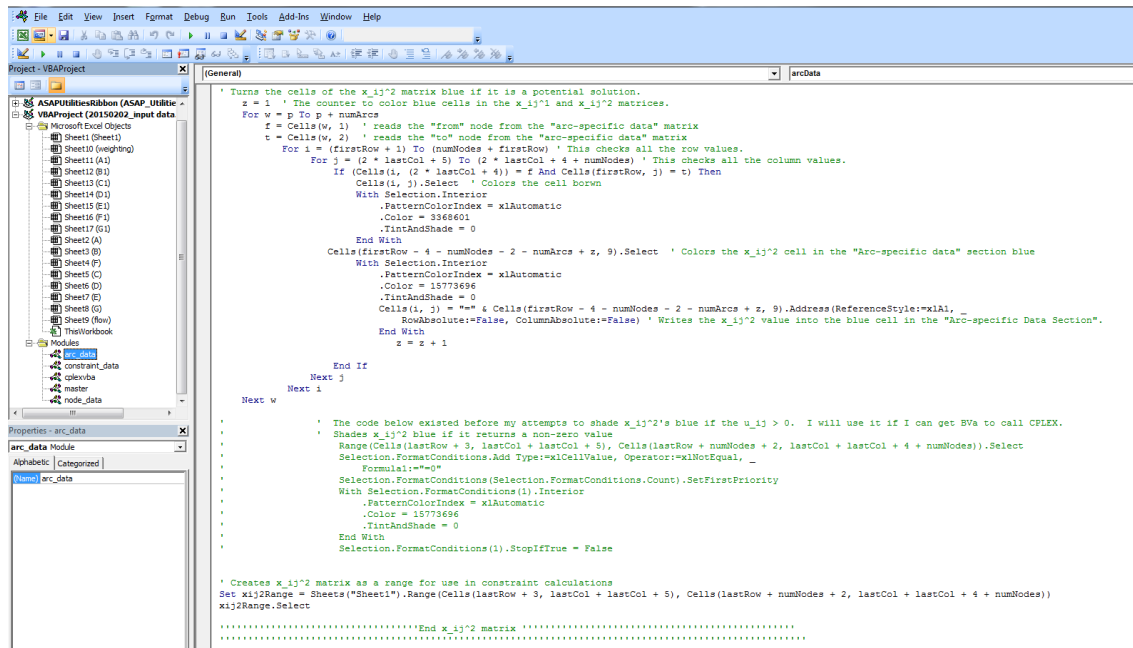
.....End c_ij^2 matrix .....

```

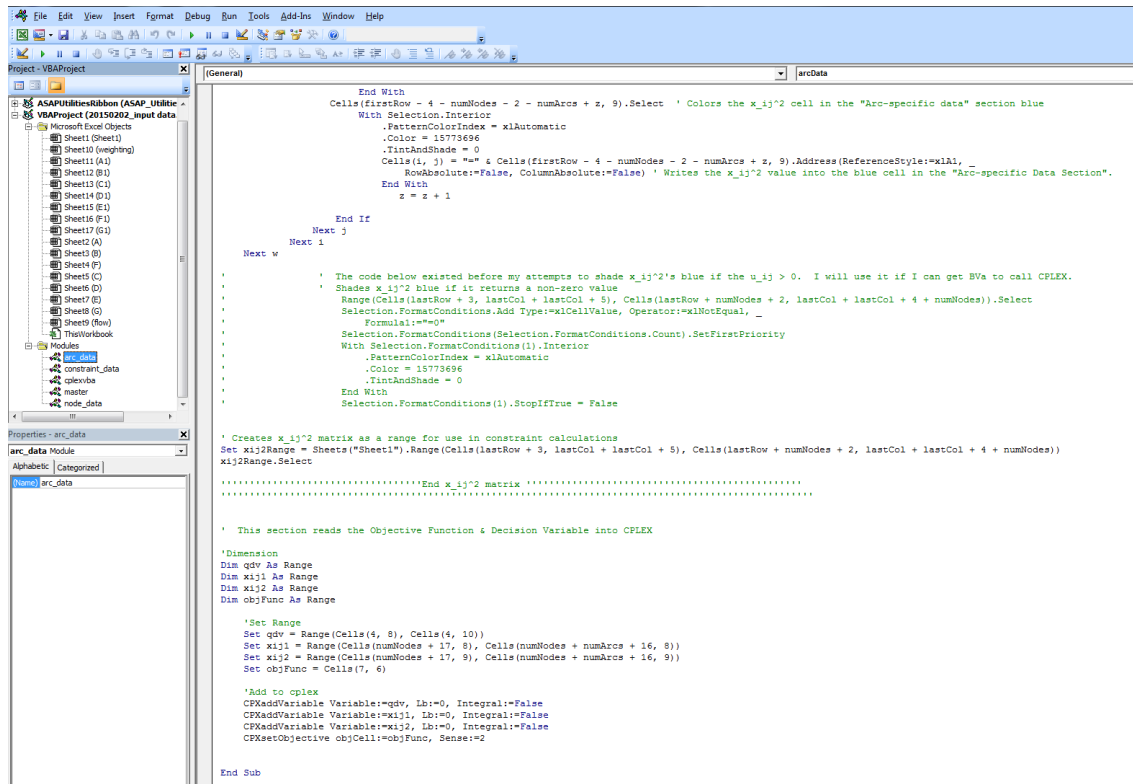
arc_data (12 of 14)



arc_data (13 of 14)



arc_data (14 of 14)



constraint_data (1 of 11)

Project - VBAProject

ASAPUtilitiesRibbon (ASAP_Utilitie
VBAProject (20150202_input data

Microsoft Excel Objects

- Sheet1 (Sheet1)
- Sheet10 (weighting)
- Sheet11 (A1)
- Sheet12 (B1)
- Sheet13 (C1)
- Sheet14 (D1)
- Sheet15 (E1)
- Sheet16 (F1)
- Sheet17 (G1)
- Sheet2 (A)
- Sheet3 (B)
- Sheet4 (F)
- Sheet5 (C)
- Sheet6 (D)
- Sheet7 (E)
- Sheet8 (G)
- Sheet9 (flow)
- ThisWorkbook

Modules

- arc_data
- constraint_data
- cplexvba
- master
- node_data

Properties - constraint_data

constraint_data Module

Alphabetic | Categorized

(Name) constraint_data

(General)

Option Explicit

```

Sub constraints()

Dim i As Integer
Dim k As Integer ' Counter for constraints (47), (48), (49), and (50)
Dim c As Integer ' Counter for constraints (47), (48), (49), and (50)
Dim j As Integer
Dim w As Integer
Dim p As Integer
Dim f As Integer
Dim t As Integer
Dim uij As Integer

'***** Finds the position of the last row*****
lastRow = ActiveSheet.Cells(Rows.Count, "b").End(xlUp).Row

' Makes the gap between 'Arc-specific data' and 'Constraints' YELLOW.
Rows(lastRow + 3).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .Color = 65535
    .TintAndShade = 0
    .PatternTintAndShade = 0
End With

' Makes the 'Arc-specific data' row RED.
Rows(lastRow + 4).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .Color = 255
    .TintAndShade = 0
    .PatternTintAndShade = 0
End With

' Writes the column heading for the "Set Matrix".
Sheets("Sheet1").Select
Range("n12").Select
ActiveCell.FormulaR1C1 = "Constraint Number"
Range("m13").Select
ActiveCell.FormulaR1C1 = "' (47) "
Range("n13").Select
ActiveCell.FormulaR1C1 = "' (48) "
Range("o13").Select
ActiveCell.FormulaR1C1 = "' (49) "
Range("p13").Select
ActiveCell.FormulaR1C1 = "' (50) "

' Makes a grid system for the Set matrix
Range("m13:p" & 13 + numNodes).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Places "0" in all of the cells of d_ij
Range(Cells(14, 13), Cells(13 + numNodes, 16)) = 0

```

constraint_data (2 of 11)

```

' Shades the first row for Set matrix.
Range("m13:p13").Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Writes the column heading "Constraints".
Cells(lastRow + 4, 2).Select
ActiveCell.FormulaR1C1 = "Constraints"
Cells(lastRow + 6, 2).Select
ActiveCell.FormulaR1C1 = "Constraint Number"

' Creates the matrix for counting constraints 47, 48, 49, 50
For i = 14 To 13 + numNodes
    ' check constraint 47
    If (Cells(i, 3).Value + Cells(i, 4).Value) = 0 Then
        Cells(i, 13).Value = 1
    Else
        ' check constraint 48
        If (Cells(i, 3).Value) = 0 And (Cells(i, 4).Value) > 0 Then
            Cells(i, 14).Value = 1
        Else
            'check constraint 49
            If (Cells(i, 3).Value) > 0 And (Cells(i, 4).Value) = 0 Then
                Cells(i, 15).Value = 1
            Else
                'check constraint 50
                If (Cells(i, 3).Value) > 0 And (Cells(i, 4).Value) > 0 Then
                    Cells(i, 16).Value = 1
                End If
            End If
        End If
    End If
Next i

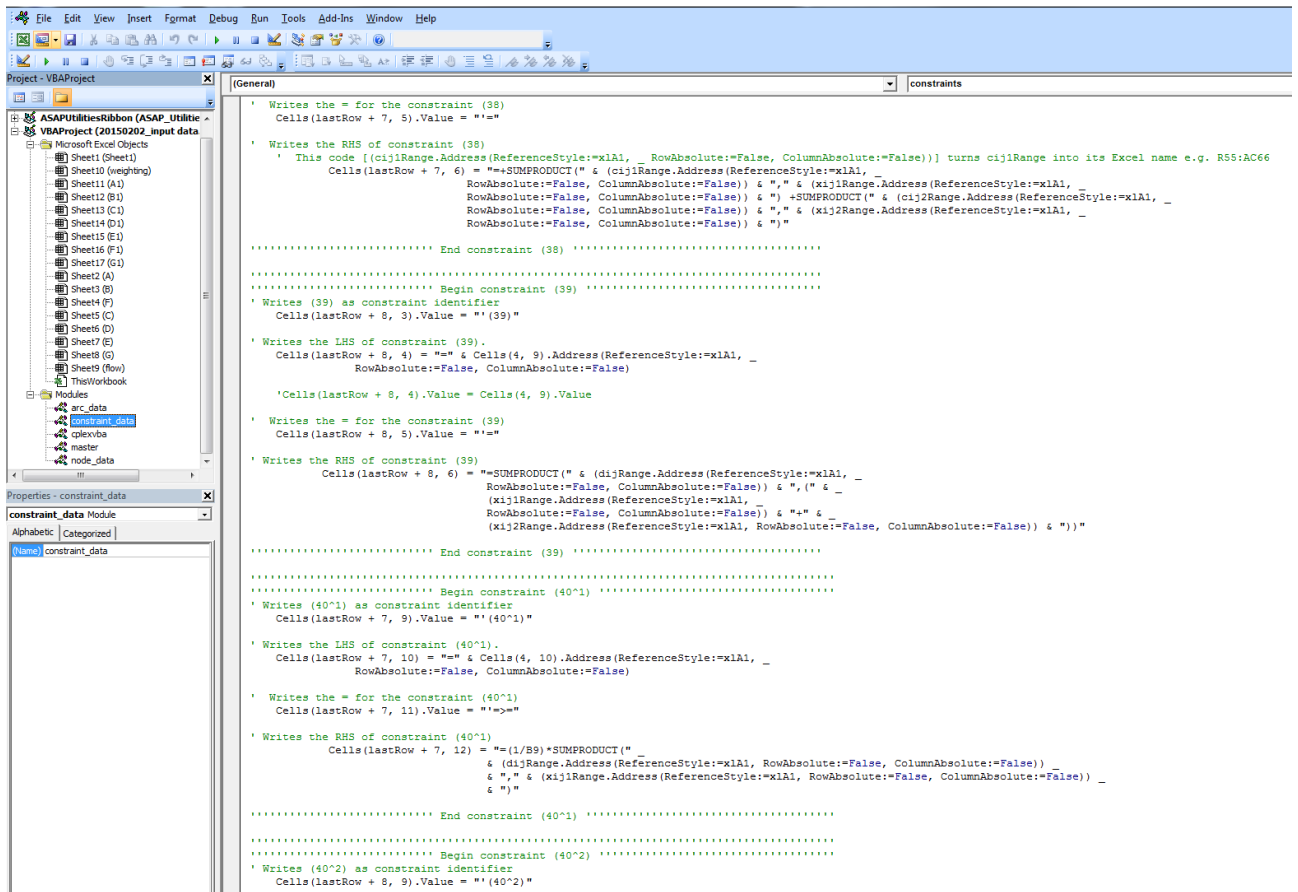
' Sum for constraints 47, 48, 49, 50 (Set matrix)
Cells(14 + numNodes, 13) = WorksheetFunction.Sum(Range(Cells(14, 13), Cells(14 + numNodes, 13)))
Cells(14 + numNodes, 14) = WorksheetFunction.Sum(Range(Cells(14, 14), Cells(14 + numNodes, 14)))
Cells(14 + numNodes, 15) = WorksheetFunction.Sum(Range(Cells(14, 15), Cells(14 + numNodes, 15)))
Cells(14 + numNodes, 16) = WorksheetFunction.Sum(Range(Cells(14, 16), Cells(14 + numNodes, 16)))
const47 = Cells(14 + numNodes, 13)
const48 = Cells(14 + numNodes, 14)
const49 = Cells(14 + numNodes, 15)
const50 = Cells(14 + numNodes, 16)

.....
' Begin constraint (38)
' Writes (38) as constraint identifier
Cells(lastRow + 7, 3).Value = "(38)"

' Writes the LHS of constraint (38).
Cells(lastRow + 7, 4) = "=" & Cells(4, 8).Address(ReferenceStyle:=xlA1, _
    RowAbsolute:=False, ColumnAbsolute:=False)

```

constraint_data (3 of 11)



constraint_data (4 of 11)

The screenshot shows the VBA Project window for a project named 'VBAPProject'. The left pane displays the project structure, including 'Microsoft Excel Objects' (Sheets 1 through 9, ThisWorkbook) and 'Modules' (arc_data, constraint_data, cplexvba, master, node_data). The 'constraint_data' module is selected. The right pane shows the VBA code for the 'constraint_data' module, which includes comments and code for writing constraint data to the worksheet.

```

' Writes the LHS of constraint (40*2).
Cells(lastRow + 8, 10) = "=" & Cells(4, 10).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False)

'Cells(lastRow + 8, 10).Value = Cells(4, 10).Value

' Writes the = for the constraint (40*2)
Cells(lastRow + 8, 11).Value = "="

' Writes the RHS of constraint (40*2)
Cells(lastRow + 8, 12) = "(1/B10)*SUMPRODUCT(" & _
& (d1jRange.Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False)) & _
& "," & (x1j2Range.Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False)) & _
& ")"

' End constraint (40*2)

' Begin constraint (41) y_i's
' Writes (41) as constraint identifier
Cells(lastRow + 7, 15).Value = "(41)"

' Writes the LHS of constraint (41).
'Range(Cells(14, 8), Cells(13 + numNodes, 8)).Select 'Used to find the column in question.
Cells(lastRow + 7, 16) = "=Sum(" & Range(Cells(14, 8), Cells(13 + numNodes, 8)) _
.Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")"

' Writes the = for the constraint (41)
Cells(lastRow + 7, 17).Value = "<="

' Writes the RHS of constraint (41)
Cells(lastRow + 7, 18) = "=" & Cells(4, 2).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False)

' End constraint (41)

' Begin constraint (42) y_i's
' Writes (42) as constraint identifier
Cells(lastRow + 8, 15).Value = "(42)"

' Writes the LHS of constraint (42).
Cells(lastRow + 8, 16) = "=Sum(" & Range(Cells(14, 7), Cells(13 + numNodes, 7)) _
.Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")"

' Writes the = for the constraint (42)
Cells(lastRow + 8, 17).Value = "<="

' Writes the RHS of constraint (42)
Cells(lastRow + 8, 18) = "=" & Cells(5, 2).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False)

' End constraint (42)

' Begin constraint (43) y_i's
' Writes (43) as constraint identifier
Cells(lastRow + 9, 15).Value = "(43)"

' Writes the LHS of constraint (43).
Cells(lastRow + 9, 16) = "=Sum(" & Range(Cells(14, 9), Cells(13 + numNodes, 9)) _
.Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")"

```

constraint_data (5 of 11)

The screenshot displays the VBA IDE for a project named 'VBAPProject'. The left-hand pane shows the 'Project - VBAPProject' window with a tree view of the project's components. Under 'Microsoft Excel Objects', there are several sheets listed. Under 'Modules', the 'constraint_data' module is selected. The right-hand pane shows the VBA code for the 'constraint_data' module. The code is written in VBA and includes comments in green. It defines constraints (43, 44, 45, 46) by writing formulas and values to specific cells in a worksheet. The code uses 'Cells' and 'Range' objects to reference cells and ranges. The constraints are defined using 'Cells' and 'Range' objects, and the code includes comments explaining the purpose of each line.

```

' Writes the = for the constraint (43)
Cells(lastRow + 9, 17).Value = "<="

' Writes the RHS of constraint (43)
Cells(lastRow + 9, 18) = "=" & Cells(6, 2).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False)

Cells(lastRow + 9, 18) = Cells(6, 2)

..... End constraint (43) .....

..... Begin constraint (44) .....

For i = 1 To numNodes
Cells(lastRow + 9 + i, 15) = "(44_" & i & ")" ' Writes (44) as a heading.
Cells(lastRow + 9 + i, 16) = "Sum(" & Range(Cells(13 + i, 7), Cells(13 + i, 9)) _
.Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
& ")" ' Writes the LHS of constraint (44).
Cells(lastRow + 9 + i, 17).Value = "<=" ' Writes <= of constraint (44)
Cells(lastRow + 9 + i, 18).Value = 1 ' Writes the RHS of constraint (44)
Next i

..... End constraint (44) .....

..... Begin constraint (45) .....

' Writes (45) as constraint identifier
For i = 1 To numNodes
Cells(lastRow + 8 + i, 3) = "(45_" & i & ")" ' Writes (45) as a heading.
Cells(lastRow + 8 + i, 4) = "Sum(" & Range(Cells(lastRow - 2 * numNodes - 1, 2 * lastCol + 4 + i), _
Cells(lastRow - numNodes - 2, 2 * lastCol + 4 + i)).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False) & ")" ' Writes the LHS of constraint (45).
Cells(lastRow + 8 + i, 5).Value = "=" ' Writes the = of constraint (45)
Cells(lastRow + 8 + i, 6) = "=" & Cells(13 + i, 10).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False) & "+" & Cells(13 + i, 11).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the RHS of constraint (45)
Next i

..... End constraint (45) .....

..... Begin constraint (46) .....

For i = 1 To numNodes
Cells(lastRow + 9 + numNodes + i, 15) = "(46_" & i & ")" ' Writes (46) as a heading.
Cells(lastRow + 9 + numNodes + i, 16) = "=" & Cells(13 + i, 10).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the LHS of constraint (46).
Cells(lastRow + 9 + numNodes + i, 17).Value = "<=" ' Writes the <= for constraint (46).
Cells(lastRow + 9 + numNodes + i, 18) = "=" & Cells(13 + i, 5).Address(ReferenceStyle:=xlA1, _
RowAbsolute:=False, ColumnAbsolute:=False) & "+" & Cells(13 + i, 9).Address _
(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the LHS of constraint (46).
Next i

..... End constraint (46) .....

```

constraint_data (6 of 11)

```

'***** Begin constraint (47) *****
k = 1
c = 0
For i = 1 To numNodes
    If Cells(13 + i, 13) = 1 Then
        Cells(lastRow + numNodes + 9 + i - k, 3) = "(47." & i & ")" ' Writes (47_i) as a constraint identifier.
        Cells(lastRow + numNodes + 9 + i - k, 4) = "=" & Cells(13 + i, 11).Address(ReferenceStyle:=xlA1, _
            RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the LHS of constraint (47).
        Cells(lastRow + numNodes + 9 + i - k, 5) = "<=" ' Writes the <= for the constraint (47)
        Cells(lastRow + numNodes + 9 + i - k, 6) = "sum(" & Range(Cells(lastRow - 2 * numNodes - 2 + k + c, 2 * lastCol + 5) _
            , Cells(lastRow - 2 * numNodes - 2 + k + c, 2 * lastCol + 4 + numNodes)) _
            .Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")" ' Writes the RHS of constraint (47)
        c = c + 1
    Else
        k = k + 1
    End If
Next i

'***** End constraint (47) *****

'***** Begin constraint (48) *****
k = 1
c = 0
For i = 1 To numNodes
    If Cells(13 + i, 14) = 1 Then
        Cells(lastRow + 2 * numNodes + 10 + i - k, 15) = "(48." & i & ")" ' Writes (48_i) as a constraint identifier.
        Cells(lastRow + 2 * numNodes + 10 + i - k, 16) = "sum(" & Range(Cells(lastRow - 2 * numNodes - 2 + i, 2 * lastCol + 5), _
            Cells(lastRow - 2 * numNodes - 2 + i, 2 * lastCol + 4 + numNodes)) _
            .Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")" ' Writes the LHS of constraint (48)
        Cells(lastRow + 2 * numNodes + 10 + i - k, 17) = "<=" ' Writes the <= for the constraint (48)
        Cells(lastRow + 2 * numNodes + 10 + i - k, 18) = "=" & Cells(13 + i, 11).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
            & "+" & Cells(13 + i, 4).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
            & "+" & Cells(13 + i, 8).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the RHS of constraint (48)
        c = c + 1
    Else
        k = k + 1
    End If
Next i

'***** End constraint (48) *****

'***** Begin constraint (49) *****
k = 1
c = 0
For i = 1 To numNodes
    If Cells(13 + i, 15) = 1 Then
        Cells(lastRow + 2 * numNodes + 10 + const48 + i - k, 15) = "(49." & i & ")" ' Writes (49_i) as a constraint identifier.
        Cells(lastRow + 2 * numNodes + 10 + const48 + i - k, 16) = "sum(" & Range(Cells(lastRow - 2 * numNodes - 2 + i, 2 * lastCol + 5), _
            Cells(lastRow - 2 * numNodes - 2 + i, 2 * lastCol + 4 + numNodes)) _
            .Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")" ' Writes the LHS of constraint (49)
        Cells(lastRow + 2 * numNodes + 10 + const48 + i - k, 17) = "<=" ' Writes the <= for the constraint (49)
        Cells(lastRow + 2 * numNodes + 10 + const48 + i - k, 18) = "=" & Cells(13 + i, 11).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
            & "+" & Cells(13 + i, 3).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
            & "+" & Cells(13 + i, 7).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the RHS of constraint (49)
        c = c + 1
    Else
        k = k + 1
    End If
Next i

```


constraint_data (7 of 11)

```

Else
    k = k + 1
End If
Next i

***** End constraint (49) *****

***** Begin constraint (50) *****

k = 1
c = 0
For i = 1 To numNodes
    If Cells(13 + i, 16) = 1 Then
        Cells(lastRow + 2 * numNodes + 10 + const48 + const49 + i - k, 15) = "[50_" & i & "]" 'Writes (50_i) as a constraint identifier.
        Cells(lastRow + 2 * numNodes + 10 + const48 + const49 + i - k, 16) = "=sum(" & Range(Cells(lastRow - 2 * numNodes - 2 + i, 2 * lastCol + 5), _
            Cells(lastRow - 2 * numNodes - 2 + i, 2 * lastCol + 4 + numNodes))
            .Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")" ' Writes the LHS of constraint (50)
        Cells(lastRow + 2 * numNodes + 10 + const48 + const49 + i - k, 17) = "<=" ' Writes the <= for the constraint (50)
        Cells(lastRow + 2 * numNodes + 10 + const48 + const49 + i - k, 18) = "=" & Cells(13 + i, 11).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
            & "+" & Cells(13 + i, 3).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
            & "+" & Cells(13 + i, 7).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
            & "+" & Cells(13 + i, 4).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
            & "+" & Cells(13 + i, 8).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the RHS of constraint (50)

        c = c + 1
    Else
        k = k + 1
    End If
Next i

***** End constraint (50) *****

***** Begin constraint (51) *****

i = 1
For i = 1 To numNodes
    Cells(lastRow + 8 + numNodes + 1 + const47, 3) = "[51_" & i & "]" ' Writes (51_i) as a constraint identifier.
    Cells(lastRow + 8 + numNodes + 1 + const47, 4) = "=sum(" & Range(Cells(lastRow - numNodes + 1, 2 * lastCol + 4 + i) _
        , Cells(lastRow - numNodes + 1, 2 * lastCol + 4 + i))
        .Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")+=" & _
        Cells(13 + i, 10).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) ' Writes the LHS of constraint (51).
    Cells(lastRow + 8 + numNodes + 1 + const47, 5) = "=" 'Writes the "=" of constraint (51).
    Cells(lastRow + 8 + numNodes + 1 + const47, 6) = "=sum(" & Range(Cells(lastRow - numNodes + 1, 2 * lastCol + 5) _
        , Cells(lastRow - numNodes + 1, 2 * lastCol + 4 + numNodes))
        .Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")+=" & _
        Cells(13 + i, 2).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
        & "+" & Cells(9, 2).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) _
        & "+" & Cells(13 + i, 9).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False) & ")" ' Writes the RHS of constraint (51).

    Next i

***** End constraint (51) *****

```

constraint_data (8 of 11)

The screenshot shows the VBA editor for a project named 'VBAProject (20150202_input data)'. The 'constraint_data' module is selected in the left-hand pane. The main window displays the following VBA code:

```

' (General)
constraints

' ***** Begin constraint (52) *****

' Writes (52) as constraint identifier
Cells(lastRow + 10 + 2 * numNodes + const48 + const49 + const50, 15).Value = "(52)"
Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50, 15).Value = "Constraint (52)"

' Writes the LHS of constraint (52).

' Places the row designation for x_ij^1 + x_ij^2 matrix
j = 1
For i = (lastRow + 13 + 2 * numNodes + const48 + const49 + const50 + 1) To _
    (lastRow + 13 + 2 * numNodes + const48 + const49 + const50 + numNodes)
    Cells(i, 15) = j
    j = j + 1
Next i

' Places the column designation for the x_ij^1 + x_ij^2 matrix
k = 1
For i = 16 To numNodes + 15 'lastCol + numNodes
    Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50 + 1, i) = k
    k = k + 1
Next i

' Makes a grid system for the x_ij^1 + x_ij^2 matrix
Range(Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50 + 1, 15), _
    Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50 + 1 + numNodes, 15 + numNodes)).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Shades the first row of the x_ij^1 + x_ij^2 matrix
Range(Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50 + 1, 15), _
    Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50 + 1, numNodes + 15)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Shades the first column of the x_ij^1 + x_ij^2 matrix
Range(Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50 + 1, 15), _
    Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50 + 1 + numNodes, 15)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670736
    .PatternTintAndShade = 0
End With

' Places "0" in all of the cells of the x_ij^1 + x_ij^2 matrix
Range(Cells(lastRow + 13 + 2 * numNodes + const48 + const49 + const50 + 1, 16), _
    Cells(lastRow + 12 + 2 * numNodes + const48 + const49 + const50 + 1 + numNodes, 15 + numNodes)) = 0

' Finds the position of the last row and last col (lastRow2 & lastCol2)
lastRow2 = ActiveSheet.Cells(Rows.Count, "o").End(xlUp).Row

```

constraint_data (9 of 11)

The screenshot shows the VBA editor for a project named 'ASAPProject (20150602_input data)'. The 'constraints' module is open, showing the following VBA code:

```

lastCol2 = Cells(lastRow2, Columns.Count).End(xlToLeft).Column

' Adds the u_ij^1 and u_ij^2 matrix
For i = 0 To (numNodes - 1)
    For j = 0 To (numNodes - 1)
        Cells(lastRow2 - numNodes + 1 + i, lastCol2 - numNodes + 1 + j) = "=" & Cells(lastRow2 - 1 - i * numNodes + 1, 2 * lastCol + 3 + j).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False)
        Cells(lastRow2 + 1 + numNodes + 1, 2 * lastCol + 5 + j).Address(ReferenceStyle:=xlA1, RowAbsolute:=False, ColumnAbsolute:=False)
    Next j
Next i

' Writes the w for the constraint (S2)
Cells(lastRow2 - 0.5 * numNodes, lastCol2 + 1) = "<="

' Writes the RMS of constraint (S2) which is the u_ij matrix from the arg_data module
' Begin u_ij matrix for the RMS of constraint (S2)

' Places the row designation for u_ij matrix
j = 1
For i = (lastRow2 - numNodes + 1) To (lastRow2)
    Cells(i, lastCol2 + 2) = j
    j = j + 1
Next i

' Places the column designation for u_ij
k = 1
For i = lastCol2 + 3 To (lastCol2 + 2 + numNodes)
    Cells(lastRow2 - numNodes, i) = k
    k = k + 1
Next i

' Makes a grid system for the u_ij matrix
Range(Cells(lastRow2 - numNodes, lastCol2 + 2), Cells(lastRow2, lastCol2 + 2 + numNodes)).Select
With Selection.Borders
    .LineStyle = xlContinuous
    .ColorIndex = 0
    .TintAndShade = 0
    .Weight = xlThin
End With

' Places "0" in all of the cells of u_ij
Range(Cells(lastRow2 - numNodes + 1, lastCol2 + 3), Cells(lastRow2, lastCol2 + 2 + numNodes)) = 0

' Shades the left column of the "u_ij" matrix
Range(Cells(lastRow2 - numNodes, lastCol2 + 2), Cells(lastRow2, lastCol2 + 2)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670796
    .PatternTintAndShade = 0
End With

' Shades the top row of the "u_ij" matrix
Range(Cells(lastRow2 - numNodes, lastCol2 + 2), Cells(lastRow2 - numNodes, lastCol2 + 2 + numNodes)).Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark1
    .TintAndShade = -0.349986266670796
    .PatternTintAndShade = 0
End With

```

The left pane shows the project structure with 'constraint_data' selected. The 'Properties: constraint_data' window is also visible, showing 'Name: constraint_data'.

constraint_data (10 of 11)

```

' Copies the u_ij matrix that was already created in the arc_data module
u1jRange.Select
Selection.Copy
Range(Cells(lastRow2 - numNodes + 1, lastCol2 + 3), Cells(lastRow2, lastCol2 + 2 + numNodes)).Select
ActiveSheet.Paste

.....

' This section reads the constraints into CPLEX

' CPLEX constraints
Dim eqConst As Range
Dim eqConstBound As Range
Dim geqConst As Range
Dim geqConstBound As Range
Dim leqConst As Range
Dim leqConstBound As Range
Dim numLeqConst As Integer
Dim numEqConst As Integer
Dim const52 As Range
Dim const52bound As Range
Dim constRow As Integer
Dim const52ind As Integer

constRow = lastRow + 7

numEqConst = 2 + 2 * numNodes + Cells(numNodes + 14, 13).Value
numLeqConst = 3 + 2 * numNodes + Application.Sum(Range(Cells(numNodes + 14, 14), Cells(numNodes + 14, 16)))

Set eqConst = Range(Cells(constRow, 4), Cells(constRow + numEqConst - 1, 4))
Set eqConstBound = Range(Cells(constRow, 6), Cells(constRow + numEqConst - 1, 6))
Set geqConst = Range(Cells(constRow, 10), Cells(constRow + 1, 10))
Set geqConstBound = Range(Cells(constRow, 12), Cells(constRow + 1, 12))
Set leqConst = Range(Cells(constRow, 16), Cells(constRow + numLeqConst - 1, 16))
Set leqConstBound = Range(Cells(constRow, 18), Cells(constRow + numLeqConst - 1, 18))

const52ind = constRow + numLeqConst - 1 + 5

Set const52 = Range(Cells(const52ind, 16), Cells(const52ind + numNodes - 1, 16 + numNodes - 1))
Set const52bound = Range(Cells(const52ind, 16 + numNodes + 2), Cells(const52ind + numNodes - 1, 16 + numNodes + 2 + numNodes - 1))

CPXaddConstraint constraint:=eqConst, Lb:=eqConstBound, Ub:=eqConstBound
CPXaddConstraint constraint:=leqConst, Ub:=leqConstBound
CPXaddConstraint constraint:=geqConst, Lb:=geqConstBound
CPXaddConstraint constraint:=const52, Ub:=const52bound
..... I might need to use the code below to copy the u_ij matrix from the arc_data module

' Places the u_ij's in the appropriate cells
For w = p To (lastRow - numNodes - 4)
    f = Cells(w, 1)
    t = Cells(w, 2)
    u1j = Cells(w, 6)
    For i = (firstRow + 1) To (numNodes + firstRow) ' This checks all the row values.
        For j = 3 To (numNodes + 2) ' This checks all the column values.
            If (Cells(i, 2) = f And Cells(firstRow, j) = t) Then
                Cells(i, j) = u1j
            End If
        Next j
    Next i
Next w

```

constraint_data (11 of 11)

The screenshot displays a VBA editor window for a project named 'VBAPProject'. The 'Project - VBAPProject' pane on the left shows the 'Modules' list with 'constraint_data' selected. The 'Properties - constraint_data' pane below it shows the 'Name' property as 'constraint_data'. The main editor pane shows the VBA code for the 'constraint_data' module, which is a CPLEX constraints module. The code defines various ranges and constants for a CPLEX model, including 'eqConst', 'eqConstBound', 'geqConst', 'geqConstBound', 'leqConst', 'leqConstBound', 'numEqConst', 'numLeqConst', 'const52', 'const52bound', and 'const52ind'. It also includes a loop to copy the 'u_ij' matrix from the 'arc_data' module.

```

'CPLEX constraints
Dim eqConst As Range
Dim eqConstBound As Range
Dim geqConst As Range
Dim geqConstBound As Range
Dim leqConst As Range
Dim leqConstBound As Range
Dim numEqConst As Integer
Dim numLeqConst As Integer
Dim const52 As Range
Dim const52bound As Range
Dim constRow As Integer
Dim const52ind As Integer

constRow = lastRow + 7

numEqConst = 2 + 2 * numNodes + Cells(numNodes + 14, 13).Value
numLeqConst = 3 + 2 * numNodes + Application.Sum(Range(Cells(numNodes + 14, 14), Cells(numNodes + 14, 16)))

Set eqConst = Range(Cells(constRow, 4), Cells(constRow + numEqConst - 1, 4))
Set eqConstBound = Range(Cells(constRow, 6), Cells(constRow + numEqConst - 1, 6))
Set geqConst = Range(Cells(constRow, 10), Cells(constRow + 1, 10))
Set geqConstBound = Range(Cells(constRow, 12), Cells(constRow + 1, 12))
Set leqConst = Range(Cells(constRow, 16), Cells(constRow + numLeqConst - 1, 16))
Set leqConstBound = Range(Cells(constRow, 18), Cells(constRow + numLeqConst - 1, 18))

const52ind = constRow + numLeqConst - 1 + 5

Set const52 = Range(Cells(const52ind, 16), Cells(const52ind + numNodes - 1, 16 + numNodes - 1))
Set const52bound = Range(Cells(const52ind, 16 + numNodes + 2), Cells(const52ind + numNodes - 1, 16 + numNodes + 2 + numNodes - 1))

CPXaddConstraint constraint:=eqConst, Lb:=eqConstBound, Ub:=eqConstBound
CPXaddConstraint constraint:=leqConst, Ub:=leqConstBound
CPXaddConstraint constraint:=geqConst, Lb:=geqConstBound
CPXaddConstraint constraint:=const52, Ub:=const52bound

' I might need to use the code below to copy the u_ij matrix from the arc_data module

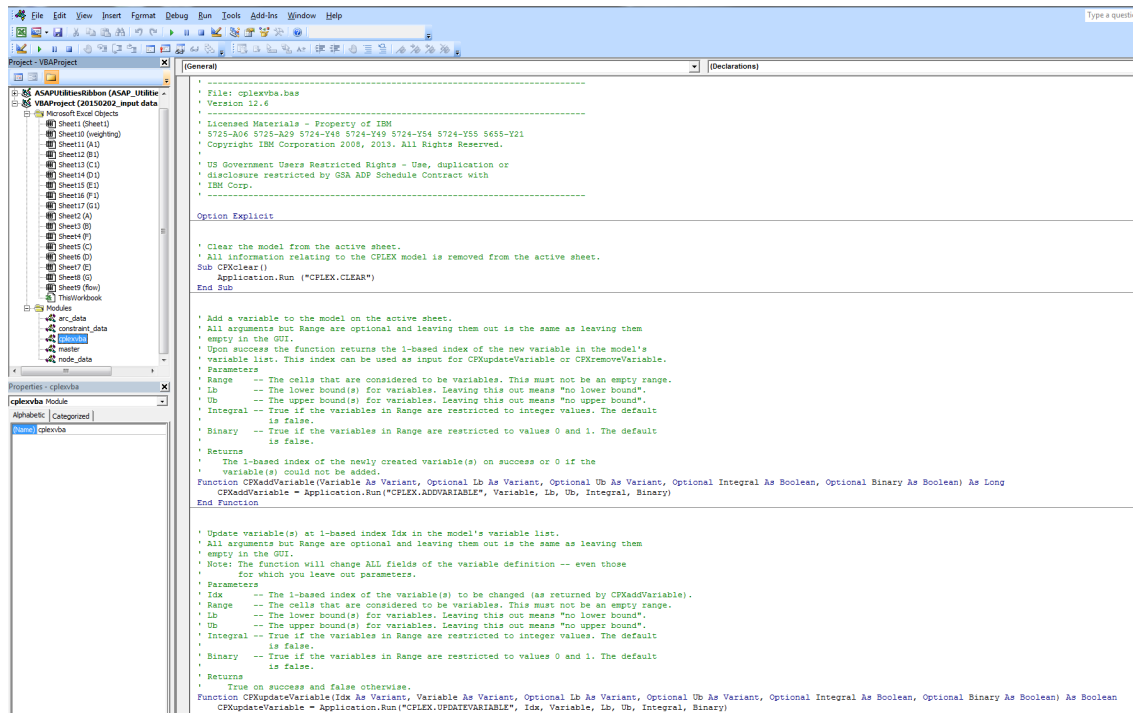
' Places the u_ij's in the appropriate cells
For w = p To (lastRow - numNodes - 4)
    f = Cells(w, 1)
    t = Cells(w, 2)
    uij = Cells(w, 6)
    For i = (firstRow + 1) To (numNodes + firstRow) ' This checks all the row values.
        For j = 3 To (numNodes + 2) ' This checks all the column values.
            If (Cells(i, 2) = f And Cells(firstRow, j) = t) Then
                Cells(i, j) = uij
            End If
        Next j
    Next i
Next w

' End u_ij matrix

' End constraint (52)
End Sub

```

cplexvba (1 of 4)



cplexvba (2 of 4)

```

End Function

' Remove variable(s) at 1-based index Idx from the model's variable list.
' Parameters
'   Idx -- The 1-based index of the variable(s) to be removed (as returned by CPXAddVariable).
'         If the argument is omitted all variables are removed from the model.
' Returns
'   True on success and false otherwise.
Function CPXremoveVariable(Optional Idx As Variant) As Boolean
    CPXremoveVariable = Application.Run("Cplex.REMOVEVARIABLE", Idx)
End Function

' Add a constraint to the model on the active sheet.
' All arguments but Range are optional and leaving them out is the same as leaving them
' empty in the GUI.
' Upon success the function returns the 1-based index of the new constraint in the model's
' constraint list. This index can be used as input for CPXupdateConstraint or
' CPXremoveConstraint.
' Parameters
'   Range -- The cells that are to be constrained. This must not be an empty range.
'   Lb     -- The lower bound(s) for the constrained cells. Leaving this out means
'             "no lower bound".
'   Ub     -- The upper bound(s) for the constrained cells. Leaving this out means
'             "no upper bound".
' Returns
'   The 1-based index of the newly created constraint on success or 0 if the
'   constraint could not be added.
Function CPXaddConstraint(constraint As Variant, Optional Lb As Variant, Optional Ub As Variant) As Long
    CPXaddConstraint = Application.Run("Cplex.ADDCONSTRAINT", constraint, Lb, Ub)
End Function

' Update constraint at 1-based index Idx in the model's constraint list.
' All arguments but Range are optional and leaving them out is the same as leaving them
' empty in the GUI.
' Note: The function will change ALL fields of the constraint definition -- even those
'       for which you leave out parameters.
' Parameters
'   Idx -- The 1-based index of the constraint to be changed (as returned by CPXaddConstraint).
'   Range -- The cells that are to be constrained. This must not be an empty range.
'   Lb     -- The lower bound(s) for the constrained cells. Leaving this out means
'             "no lower bound".
'   Ub     -- The upper bound(s) for the constrained cells. Leaving this out means
'             "no upper bound".
' Returns
'   True on success and false otherwise.
Function CPXupdateConstraint(Idx As Variant, constraint As Variant, Optional Lb As Variant, Optional Ub As Variant) As Boolean
    CPXupdateConstraint = Application.Run("Cplex.UPDATECONSTRAINT", Idx, constraint, Lb, Ub)
End Function

' Remove constraint at 1-based index Idx from the model's variable list.
' Parameters
'   Idx -- The 1-based index of the constraint to be removed (as returned by CPXaddConstraint).
'         If the argument is omitted all constraints are removed from the model.
' Returns
'   True on success and false otherwise.
Function CPXremoveConstraint(Optional Idx As Variant) As Boolean
    CPXremoveConstraint = Application.Run("Cplex.REMOVECONSTRAINT", Idx)
End Function

```

cplexvba (3 of 4)

```

' Set the CPLEX Excel-specific parameters.
' Parameters left out in the argument list will not be changed.
' Parameters
' StopInt -- Flag to control whether to stop at each integral solution found.
' LinOrQuad -- Flag to control whether to display a warning message if CPLEX
'              found an unknown function and replaced that with its presumed linear
'              or quadratic equivalent.
' ExportModel -- Flag to control whether the optimization model is to be exported
'              for debugging purposes.
' ModelFile -- Name of file to which model is exported if ExportModel is true.
' Returns
' True on success and false otherwise.
Function CPXsetSpecial(Optional StopInt As Variant, Optional LinOrQuad As Variant, Optional ExportModel As Variant, Optional ModelFile As Variant) As Boolean
    CPXsetSpecial = Application.Run("CPLEX.SETSPECIAL", StopInt, LinOrQuad, ExportModel, ModelFile)
End Function

' Add a parameter to the parameter list on the active spreadsheet's model.
' Note: The function does not check for duplicate parameters in the parameter list
'       of the model.
' The index returned by this function can be used as argument to CPXupdateParameter or
' CPXremoveParameter.
' Parameters
' Number -- The number or name of the parameter to be set.
' Value -- The value to be set for parameter Number.
' Returns
' The 1-based index of the newly created parameter in the model's parameter list on
' success and 0 on failure.
Function CPXaddParameter(Number As Variant, Value As Variant) As Boolean
    CPXaddParameter = Application.Run("CPLEX.ADDPARAMETER", Number, Value)
End Function

' Update parameter at 1-based index Idx.
' Parameters
' Idx -- The 1-based index of the parameter to be changed (as returned by CPXaddParameter).
' Number -- The number or name of the parameter to be set.
' Value -- The value to be set for parameter Number.
' Returns
' True on success and false otherwise.
Function CPXupdateParameter(Index As Variant, Number As Variant, Value As Variant) As Boolean
    CPXupdateParameter = Application.Run("CPLEX.UPDATEPARAMETER", Index, Number, Value)
End Function

' Remove parameter at 1-based index Idx from parameter list.
' Idx -- The 1-based index of the parameter to be removed (as returned by CPXaddParameter).
'       If omitted all parameters are removed.
' Returns
' True on success and false on failure.
Function CPXremoveParameter(Index As Variant) As Boolean
    CPXremoveParameter = Application.Run("CPLEX.REMOVEPARAMETER", Index)
End Function

' Set the objective function for the model on the active sheet.
' Target is only optional if Sense is not 3.
' Parameters
' ObjCell -- The objective function cell. This must be a single cell.
' Sense -- The sense of the objective function. Allowed values are
'           1 - Maximize ObjCell
'           2 - Minimize ObjCell

```


cplexvba (4 of 4)

```

CPXaddParameter = Application.Run("CPLEX.ADDPARAMETER", Number, Value)
End Function

' Update parameter at 1-based index Idx.
' Parameters
' Idx -- The 1-based index of the parameter to be changed (as returned by CPXaddParameter).
' Number -- The number or name of the parameter to be set.
' Value -- The value to be set for parameter Number.
' Returns
' True on success and false otherwise.
Function CPXupdateParameter(Index As Variant, Number As Variant, Value As Variant) As Boolean
    CPXupdateParameter = Application.Run("CPLEX.UPDATEPARAMETER", Index, Number, Value)
End Function

' Remove parameter at 1-based index Idx from parameter list.
' Idx -- The 1-based index of the parameter to be removed (as returned by CPXaddParameter).
' If omitted all parameters are removed.
' Returns
' True on success and false on failure.
Function CPXremoveParameter(Index As Variant) As Boolean
    CPXremoveParameter = Application.Run("CPLEX.REMOVEPARAMETER", Index)
End Function

' Set the objective function for the model on the active sheet.
' Target is only optional if Sense is not 3.
' Parameters
' ObjCell -- The objective function cell. This must be a single cell.
' Sense -- The sense of the objective function. Allowed values are
'         1 - Maximize ObjCell
'         2 - Minimize ObjCell
'         3 - Find a solution so that ObjCell equals Target
' Target -- The target value for Sense = 3.
' Returns
Function CPXsetObjective(objCell As Variant, Sense As Integer, Optional Target As Double) As Boolean
    CPXsetObjective = Application.Run("CPLEX.SETOBJECTIVE", objCell, Sense, Target)
End Function

' Invokes CPLEX on the model defined on the active sheet.
' Note: If this function returns true that does not mean that a solution was found.
'       CPLEX may also have successfully proved infeasibility of the model.
' NoDialog -- If a true value is passed the final dialog will not be shown. In this case
'             there is no way to reset the variable cells to their original values
'             or perform a solution or sensitivity analysis.
'             If the value passed cannot be converted to a boolean it is ignored.
' Returns
' True on success and false on failure.
Function CPXsolve(Optional NoDialog As Variant) As Boolean
    CPXsolve = Application.Run("CPLEX.SOLVE", NoDialog)
End Function

' Displays the CPLEX Excel connector's solve dialog.
Sub CPXdialog()
    Application.Run ("CPLEX.DIALOG")
End Sub

```

The Military Theater Distribution Network Design Problem



Advisor: LTC Brian J. Lunday, PhD

Readers: Dr. Ray R. Hill, PhD, MAJ Peter Nesbitt

Department of Operational Sciences (ENS)

Air Force Institute of Technology

Sponsor:
TRAC-LEE

Objectives:

- Develop a math programming model to locate aerial/sea ports of debarkation (A/SPOD) and distribution centers (DC) in order to maximize multi-commodity flows in support of contingency operations.
- Develop a simple-to-use-and-understand user input-process.
- Demonstrate the model by using a scenario-based case study.

Background:

- The movement of supplies into an Area of Operations is a joint endeavor.
- The US Army is responsible for the ground movement of supplies from A/SPODs to the force in the field.
- The Army distribution hierarchy supports the operational hierarchy to distribute supplies.

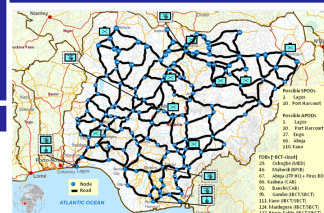
Methodology:

- Move supplies through a transportation network as Mixed Integer Linear Program
- Locate A/SPODs and DCs with multi-objective approach to meet FOB demands
- The multi-objective approach includes:
 - Risk to supplies being transported.
 - Total distance travelled by supplies
 - Maximum per capita workload supported by transportation assets at a given echelon
- Classify flow as either A/SPOD to DC (Echelon 1) or DC to FOB (Echelon 2)
- Use echelon flow to identify routing

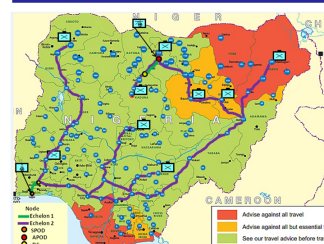
Network:

- **Nodes:** 135 road junctions across Nigeria
- **Arcs:** 414 arcs which are the roads connecting the 135 nodes
- **Arcs:** Classified by their supply throughput capacity

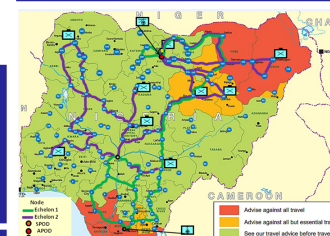
Network Depiction



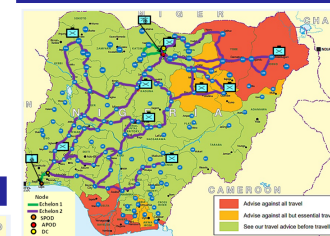
Optimal Solution for Minimal Risk



Optimal Solution for Maximum per capita workload

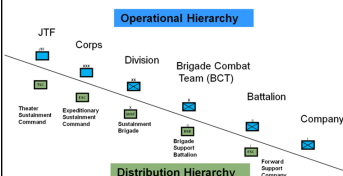


Optimal Solution for Minimizing Total Supply Distance



Conclusions:

- The model handles multiple objectives and user priorities to locate A/SPODs and DCs
- The simple-to-use-and-understand user input-process solves realistic sized instances within two minutes
- The model uses a weighting scheme to allow decision maker influence



Literature Review pertaining to:

- Facility Location Models
- Network Flow Models
- Location and Routing Models

[illegible]

Bibliography

1. Ahuja, Ravindra K, Magnanti, Thomas L, & Orlin, James B. 1993. Network flows: theory, algorithms, and applications.
2. Aykin, Turgut. 1995. The hub location and routing problem. *European Journal of Operational Research*, **83**(1), 200–219.
3. Balachandran, M, & Gero, JS. 1984. A comparison of three methods for generating the Pareto optimal set. *Engineering Optimization*, **7**(4), 319–336.
4. Beasley, JE, & Christofides, Nicos. 1989. An algorithm for the resource constrained shortest path problem. *Networks*, **19**(4), 379–394.
5. Bellman, Richard. 1956. *On a routing problem*. Tech. rept. DTIC Document.
6. Boccia, Maurizio, Crainic, Teodor G, Sforza, Antonio, & Sterle, Claudio. 2010. A metaheuristic for a two echelon location-routing problem. *Pages 288–301 of: Experimental Algorithms*. Springer.
7. Chairman of the Joint Chiefs of Staff. 2013. Joint Publication 4-0 Logistics.
8. Church, Richard, & Velle, Charles R. 1974. The maximal covering location problem. *Papers in regional science*, **32**(1), 101–118.
9. Command and General Staff College. 2013. Theater Sustainment Battle Book. *Fort Leavenworth, KS*.
10. Contardo, Claudio, Hemmelmayr, Vera, & Crainic, Teodor Gabriel. 2012. Lower and upper bounds for the two-echelon capacitated location-routing problem. *Computers & Operations Research*, **39**(12), 3185–3199.
11. Current, John, Ratick, Samuel, & ReVelle, Charles. 1998. Dynamic facility location when the total number of facilities is uncertain: A decision analysis approach. *European Journal of Operational Research*, **110**(3), 597–609.
12. Dantzig, George B. 1963. *Linear programming and its extensions*. Princeton University Press, Princeton, NJ.
13. Dantzig, George Bernard, Ford Jr, Lester Randolph, & Fulkerson, Delbert Ray. 1956. *A PRIMAL–DUAL ALGORITHM*. Tech. rept. DTIC Document.
14. Daskin, Mark S. 2011. *Network and discrete location: models, algorithms, and applications*. John Wiley & Sons.
15. Department of the Army. 2012. ADP 4-0 Sustainment. *Washington:DOA*.

16. Department of the Army. 2013. Theater Sustainment Command. *Washington:DOA*.
17. Dijkstra, Edsger W. 1959. A note on two problems in connexion with graphs. *Numerische mathematik*, **1**(1), 269–271.
18. Edmonds, Jack, & Karp, Richard M. 1972. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, **19**(2), 248–264.
19. Fulkerson, D Ray, & Harding, Gary C. 1977. Maximizing the minimum source-sink path subject to a budget constraint. *Mathematical Programming*, **13**(1), 116–118.
20. Geoffrion, Arthur M, & Graves, Glenn W. 1974. Multicommodity distribution system design by Benders decomposition. *Management Science*, **20**(5), 822–844.
21. Ginn, J. 2014. *Personal Interview, Major Ginn, US Army, has served in Army logistics units for over 14 years at both the strategic and tactical levels, including planning support to operations in Iraq and Afghanistan*. Tech. rept. US Army.
22. Hakimi, S Louis. 1964. Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations research*, **12**(3), 450–459.
23. Hamacher, Horst W, & Drezner, Zvi. 2002. *Facility location: applications and theory*. Springer.
24. Hillier, Frederick S, & Lieberman, Gerald J. 2001. *Introduction to operations research*. Tata McGraw-Hill Education.
25. International Military Forums. 2012. *Logistics Quotes*. Tech. rept. UK Foreign Travel. <http://www.military-quotes.com/forum/logistics-quotes-t511.html>, Accessed:2015-01-19.
26. Jackson, Jack. 2007. Logistics Battle Command. *MORS 75th Annual Symposium*.
27. Jacobsen, S Kruse, & Madsen, Oli BG. 1980. A comparative study of heuristics for a two-level routing-location problem. *European Journal of Operational Research*, **5**(6), 378–387.
28. Lamothe, Dan. 2008. Former Commandant Barrow Dies at 86. *The Marine Corps Times*.
29. Lin, Jenn-Rong, & Lei, Hsien-Chung. 2009. Distribution systems design with two-level routing considerations. *Annals of Operations Research*, **172**(1), 329–347.
30. Madsen, Oli BG. 1983. Methods for solving combined two level location-routing problems of realistic dimensions. *European Journal of Operational Research*, **12**(3), 295–301.

31. Map, Nigeria Road. 2012. *Maps of World*. Tech. rept. Nigeria Road Map. <http://www.mapsofworld.com/nigeria/road-map.html>, Accessed:2015-01-19.
32. Mumphrey, Anthony J, Seley, John E, & Wolpert, Julian. 1971. A decision model for locating controversial facilities. *Journal of the American Institute of Planners*, **37**(6), 397–402.
33. Nagy, Gábor, & Salhi, Saïd. 2007. Location-routing: Issues, models and methods. *European Journal of Operational Research*, **177**(2), 649–672.
34. Nguyen, Viet-Phuong, Prins, Christian, & Prodhon, Caroline. 2012. A multi-start iterated local search with tabu list and path relinking for the two-echelon location-routing problem. *Engineering Applications of Artificial Intelligence*, **25**(1), 56–71.
35. Perl, Jossef, & Daskin, Mark S. 1985. A warehouse location-routing problem. *Transportation Research Part B: Methodological*, **19**(5), 381–396.
36. Prodhon, Caroline, & Prins, Christian. 2014. A survey of recent research on location-routing problems. *European Journal of Operational Research*, **238**(1), 1–17.
37. Tcha, Dong-wan, & Lee, Bum-il. 1984. A branch-and-bound algorithm for the multi-level uncapacitated facility location problem. *European Journal of Operational Research*, **18**(1), 35–43.
38. Toregas, Constantine, Swain, Ralph, ReVelle, Charles, & Bergman, Lawrence. 1971. The location of emergency service facilities. *Operations Research*, **19**(6), 1363–1373.
39. United Kingdom Foreign & Commonwealth Office. 2012. *FCO Updates Nigeria Travel Data*. Tech. rept. UK Foreign Travel. <http://news.carrentals.co.uk/fco-updates-nigeria-travel-advice-34256830.html>, Accessed:2015-01-19.
40. United States Department of the Army. 2011. ATTP 5-0.1 Commander and Staff Officer Guide. *Washington:DOA*.
41. Verter, Vedat. 2011. Uncapacitated and capacitated facility location problems. *Pages 25–37 of: Foundations of Location Analysis*. Springer.
42. Webb, MHJ. 1968. Cost functions in the location of depots for multiple-delivery journeys. *Operations Research Quarterly*, 311–320.
43. World Food Programme. 2012. *Nigeria Road Network*. Tech. rept. World Food Programme. <http://dlca.logcluster.org/display/public/DLCA/2.3+Nigeria+Road+Network;jsessionid=4202ABE1E9CE6CBE1B8A67213A8C3D93>, Accessed:2015-01-19.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE (DD-MM-YYYY) 26-03-2015			2. REPORT TYPE Master's Thesis		3. DATES COVERED (From — To) Oct 2013 — Mar 2015	
4. TITLE AND SUBTITLE The Military Theater Distribution Network Design Problem					5a. CONTRACT NUMBER	
					5b. GRANT NUMBER	
					5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Craig, Robert R., Major, USA					5d. PROJECT NUMBER	
					5e. TASK NUMBER	
					5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/ENS) 2950 Hobson Way WPAFB OH 45433-7765					8. PERFORMING ORGANIZATION REPORT NUMBER AFIT-ENS-MS-15-M-137	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Morris Hayes TRADOC Analysis Center, Ft. Lee 700 Quarters Suite 210 Ft. Lee, VA 23801-1703 Email: morris.g.hayes.civ@mail.mil					10. SPONSOR/MONITOR'S ACRONYM(S) TRAC-LEE	
					11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION / AVAILABILITY STATEMENT Distribution Statement A: Approved for Public Release; distribution unlimited.						
13. SUPPLEMENTARY NOTES This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.						
14. ABSTRACT We formulate and test a mathematical program to select Air and Sea Ports of Debarkation and intermediate logistical distribution centers, through which we route military supplies over a directed transportation network to meet aggregated weekly demands by military units conducting a steady state contingency operation. The multi-objective model seeks to minimize a weighted combination of the total risk encountered by transported supplies, the total distance traveled by supplies, and the maximum per capita workload supported by transportation assets at a given echelon (i.e., port-to-distribution center versus distribution center to demands). Within our formulation, we account for capacities on arc flows and node throughputs, with the latter enabling the representation of material handling equipment limitations at the ports and distribution centers. For our model, we develop and test an Excel-based decision support tool that invokes the commercial solver CPLEX (Version 12.5) to solve the underlying mixed-integer linear program, and we demonstrate its efficacy on a representative instance. For this instance, we identify extreme points and selected interior solutions on the Pareto efficient frontier and examine the model's sensitivity to selected parameters. We conclude by discussing how the model can account for intra-theater airlift and outline modifications that can account for expected pilferage losses within the distribution system						
15. SUBJECT TERMS Network; Facility Location; Supply Chain Design						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON	
a. REPORT	b. ABSTRACT	c. THIS PAGE			LTC Brian J. Lunday, (ENS)	
U	U	U	UU	122	19b. TELEPHONE NUMBER (include area code) (937) 255-3636, x4555; brian.lunday@afit.edu	